RECOVER PLUS for DB2®
Reference Manual

Supporting

Version 10.1 of RECOVER PLUS for DB2
Version 10.1 of Administrative Assistant for DB2
Version 10.1 of Database Administration for DB2
Version 10.1 of Recovery Management for DB2

April 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 | 713 918 8800 or 800 841 2031 | 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>


BMC, BMC Software, and the BMC Software logo are the exclusive properties of BMC Software, Inc., are registered with the U.S. Patent and Trademark Office, and may be registered or pending registration in other countries. All other BMC trademarks, service marks, and logos may be registered or pending registration in the U.S. or in other countries. All other trademarks or registered trademarks are the property of their respective owners.

DB2 is a registered trademark of International Business Machines Corporation.

DB2, IBM, MVS, OS/390, RACF, SQL/DS, System z, z/OS, and zSeries are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

The information included in this documentation is the proprietary and confidential information of BMC Software, Inc., its affiliates, or licensors. Your use of this information is subject to the terms and conditions of the applicable End User License agreement for the product and to the proprietary and restricted rights notices included in the product documentation.

**Restricted rights legend**

U.S. Government Restricted Rights to Computer Software. UNPUBLISHED -- RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES. Use, duplication, or disclosure of any data and computer software by the U.S. Government is subject to restrictions, as applicable, set forth in FAR Section 52.227-14, DFARS 252.227-7013, DFARS 252.227-7014, DFARS 252.227-7015, and DFARS 252.227-7025, as amended from time to time. Contractor/Manufacturer is BMC SOFTWARE INC, 2101 CITYWEST BLVD, HOUSTON TX 77042-2827, USA. Any contract notices should be sent to this address.
Customer support

You can obtain technical support by using the BMC Software Customer Support 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 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
- Related documentation ........................................... 22
- Conventions ...................................................... 23
- Syntax diagrams ............................................... 24
- Summary of changes .......................................... 26

Chapter 1 Introducing RECOVER PLUS 43
- Overview ....................................................... 44
- Solution integration .......................................... 45
- Features and benefits of RECOVER PLUS ............... 46
  - Reduced elapsed time required to recover .......... 46
  - Better audit and recovery process control ........ 48
  - Expanded recovery capabilities ...................... 48
  - Recovery test capability ................................ 50
  - TRANSFORM and High-speed Structure Change .... 50
- RECOVER PLUS operation ................................. 51
  - Using IBM DB2 resources ................................ 53
  - Using RECOVER PLUS with RECOVERY MANAGER ... 54
  - Using RECOVER PLUS with R+/CHANGE ACCUM .... 54
- RECOVER PLUS limitations ............................... 55
- Planning recovery ........................................... 57
- Recovery strategies ......................................... 58
  - Using LOGSCAN for estimates ......................... 58
  - Using BACKOUT for point-in-time recovery without image copies 59
  - Comparing BACKOUT performance ................ 60
  - Using BACKOUT AUTO recovery ..................... 61
  - Using copies and logs for index recovery ........... 62
  - Performing forward recovery without an image copy 62
  - Recovering with Instant Snapshot copies .......... 62
  - Using parallel MERGE phases and log sorts ....... 63
  - Using multitasking index key sorts and index rebuilds 63
  - Eliminating interim data sets for index keys ...... 64
  - Rebuilding nonpartitioned indexes on a partitioned table space 64
  - Using resource selection ................................ 64
  - Previewing recovery activity ......................... 65
  - Testing recovery with simulation ..................... 66
  - Recovering while activity continues ................ 68
  - BMCXCOPY registrations ................................. 68
  - Creating copies from recovery resources .......... 69
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overriding registered copies</td>
<td>69</td>
</tr>
<tr>
<td>Recovering LOB spaces with RECOVER PLUS</td>
<td>69</td>
</tr>
<tr>
<td>Automatic fallback without restarting</td>
<td>71</td>
</tr>
<tr>
<td>Using incremental index image copies</td>
<td>71</td>
</tr>
<tr>
<td>Using cabinet copies</td>
<td>73</td>
</tr>
<tr>
<td>Using Online Consistent Copies</td>
<td>75</td>
</tr>
<tr>
<td>Solving other problems with RECOVER PLUS</td>
<td>76</td>
</tr>
<tr>
<td>Chapter 2 Operational considerations</td>
<td>77</td>
</tr>
<tr>
<td>Operating environment</td>
<td>78</td>
</tr>
<tr>
<td>DB2 support</td>
<td>78</td>
</tr>
<tr>
<td>System requirements</td>
<td>78</td>
</tr>
<tr>
<td>Setting the MEMLIMIT parameter</td>
<td>79</td>
</tr>
<tr>
<td>Software requirements</td>
<td>80</td>
</tr>
<tr>
<td>Authorization</td>
<td>80</td>
</tr>
<tr>
<td>DB2 authority</td>
<td>80</td>
</tr>
<tr>
<td>System authority</td>
<td>81</td>
</tr>
<tr>
<td>APF authority</td>
<td>82</td>
</tr>
<tr>
<td>Access control authorization exit.</td>
<td>82</td>
</tr>
<tr>
<td>Table space and index space status</td>
<td>83</td>
</tr>
<tr>
<td>Initial table space and index space status</td>
<td>84</td>
</tr>
<tr>
<td>Restoring initial status</td>
<td>86</td>
</tr>
<tr>
<td>Serialization and concurrency issues</td>
<td>86</td>
</tr>
<tr>
<td>Concurrency with IBM DB2 utilities</td>
<td>86</td>
</tr>
<tr>
<td>Concurrency with other BMC utilities for DB2</td>
<td>87</td>
</tr>
<tr>
<td>Concurrency with other RECOVER PLUS executions</td>
<td>89</td>
</tr>
<tr>
<td>Recovery using a non-DB2 backup</td>
<td>90</td>
</tr>
<tr>
<td>Recovery using cumulative incremental image copies</td>
<td>90</td>
</tr>
<tr>
<td>Checkpoints for RECOVER PLUS restart</td>
<td>91</td>
</tr>
<tr>
<td>Checkpoint installation options</td>
<td>91</td>
</tr>
<tr>
<td>Checkpoint override parameter</td>
<td>92</td>
</tr>
<tr>
<td>Storage group-defined data sets</td>
<td>92</td>
</tr>
<tr>
<td>Primary space allocation</td>
<td>93</td>
</tr>
<tr>
<td>Secondary extents</td>
<td>94</td>
</tr>
<tr>
<td>Chapter 3 RECOVER PLUS syntax</td>
<td>95</td>
</tr>
<tr>
<td>Overview</td>
<td>96</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>96</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>97</td>
</tr>
<tr>
<td>RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE</td>
<td>97</td>
</tr>
<tr>
<td>REBUILD INDEX</td>
<td>98</td>
</tr>
<tr>
<td>RECOVER UNLOADKEYS</td>
<td>98</td>
</tr>
<tr>
<td>RECOVER BUILDINDEX</td>
<td>99</td>
</tr>
<tr>
<td>LOGSCAN</td>
<td>99</td>
</tr>
<tr>
<td>Adding comments</td>
<td>99</td>
</tr>
<tr>
<td>Use of long names</td>
<td>100</td>
</tr>
<tr>
<td>Support for Unicode names</td>
<td>101</td>
</tr>
<tr>
<td>Alphabetical listing of RECOVER PLUS options</td>
<td>102</td>
</tr>
</tbody>
</table>
Chapter 4  Building and executing RECOVER PLUS jobs  267

Overview .......................... 268
Building RECOVER PLUS jobs .......... 268
  JOB statement .................. 268
  EXEC statement ................. 269
  REGION parameter .......... 269
  Utility parameters on the EXEC statement .......................... 269
  STEPLIB DD or JOBLIB statements .......... 276
  RECOVER PLUS data sets and RECOVER PLUS DD statements .......... 276
  Defining sort work data sets .................. 286
  Determining sort work space requirements .......... 287
Determining unloaded keys file space requirements .................................................. 290
Specifying RECOVER PLUS syntax ............................................................................ 291
Running RECOVER PLUS jobs .................................................................................... 291
Starting a RECOVER PLUS job .................................................................................... 291
Restarting a RECOVER PLUS job ................................................................................ 292
Displaying the status of a RECOVER PLUS job .............................................................. 297
Terminating a RECOVER PLUS job .............................................................................. 297
Cleaning up the BMCUTIL and BMCSYNC tables ......................................................... 297
Cleaning up after a RECOVER UNLOADKEYS job ......................................................... 297
Using MSGLEVEL to control RECOVER PLUS output ................................................. 298
Sample output with MSGLEVEL(0) ............................................................................. 298
Sample output with MSGLEVEL(1) ............................................................................. 301
Sample output with MSGLEVEL(2) ............................................................................. 304
Examples of RECOVER PLUS operations .................................................................... 311
Multiple table space recovery and index rebuilds ......................................................... 311
Parallel index key sorts and multitasking index rebuilds ............................................... 312
Multiple log sorts, MERGE phases, and parallel index key sorts with multitasking index rebuilds ................................................................. 314
Multiple table space and index recoveries ................................................................... 316
BACKOUT recovery ...................................................................................................... 317
Instant Snapshot recovery ............................................................................................. 318

Chapter 5 Examples of RECOVER PLUS jobs 319

Overview ......................................................................................................................... 320
Example 1: Previewing the recovery plan .......................................................... 325
Example 2: Recovering to the current state ........................................................ 334
  Example 2A: Using REBUILD INDEX ...................................................................... 334
  Example 2B: Using RECOVER INDEX ..................................................................... 340
Example 3: Creating and dynamically allocating output copies ...................... 359
Example 4: Recovering to a specified copy ........................................................ 360
  Example 4A: Using TOCOPY with a named copy .............................................. 360
  Example 4B: Using TOCOPY LASTCOPY ........................................................... 360
Example 5: Recovering to a specific log point .................................................... 363
  Example 5A: Using the TOLOGPOINT option with RECOVER ......................... 363
  Example 5B: Using TOLOGPOINT and BACKOUT ........................................... 364
Example 6: Extracting nonpartitioned index keys .................................................. 365
Example 7: Building a nonpartitioned index ............................................................. 366
Example 8: Recovering using tape-stacked data sets ............................................ 367
  Example 8A: Making stacked image copies ...................................................... 367
  Example 8B: Recovering with stacked image copies ....................................... 368
Example 9: Recovering to a non-DB2 data set ......................................................... 369
  Directing recovery output ......................................................................................... 369
Example 10: Overriding installation options .............................................................. 372
  Example 10A: Overriding several installation options .................................. 372
  Example 10B: Overriding the IXRECP installation option .............................. 373
Example 11: Recovering using a change accumulation file .................................. 383
  Example 11A: Creating the change accumulation file .................................. 383
  Example 11B: Using the change accumulation file ....................................... 383
Example 12: Using the MAINT parameter ............................................................... 384
<table>
<thead>
<tr>
<th>Example</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 13: Using the LOGSCAN command</td>
<td>387</td>
</tr>
<tr>
<td>Example 14: Simulating recovery</td>
<td>390</td>
</tr>
<tr>
<td>Example 15: Using MAXKSORT for parallel index rebuilds</td>
<td>391</td>
</tr>
<tr>
<td>Example 16: Using MAXKSORT and recovering a table space</td>
<td>397</td>
</tr>
<tr>
<td>Example 17: Using timestamp recovery</td>
<td>398</td>
</tr>
<tr>
<td>Example 18: Using the MAXLSORT default value</td>
<td>399</td>
</tr>
</tbody>
</table>

**Chapter 6 RECOVER PLUS concepts**

- Overview ............................................. 410
- Recovery of partitioned table spaces and indexes .......................... 410
  - Recovering multiple table space partitions in separate jobs .......... 411
  - Recovering table space partitions and rebuilding a partitioned index in multiple jobs ......................... 413
  - Rebuilding nonpartitioned indexes with single and multiple jobs ........ 414
- Recovering a partitioned table space and rebuilding a nonpartitioned index ................................................. 416
  - Recovering a table space and rebuilding a partitioned index and a nonpartitioned index .................. 417
- Understanding the LOGSORT strategy .................................... 418
  - Why is it faster to sort the log? ........................................ 418
  - What are the restrictions when using LOGSORT? ..................... 419
  - Does LOGSORT have other advantages or disadvantages? ............ 419
- Using LOGONLY or LOGAPPLY ONLY with point-in-time recoveries ........ 421
- Understanding the NOWORKDDN strategy ................................. 422
- Understanding the UNLOADKEYS and BUILDINDEX strategy ............... 423
- Using SHRLEVEL CHANGE full copies for recovery ...................... 424
  - Attempting to recover to a SHRLEVEL CHANGE copy ................ 424
- Fallback recovery with RECOVER PLUS .................................... 424
- Strategies for point-in-time recovery .................................. 426
  - Using BACKOUT ............................................ 426
  - Using Instant Snapshots ........................................ 429
- Working with ASCII and Unicode data ..................................... 433
- Managing sort performance ................................................ 433
  - Recommendations for a large number of concurrent sorts ............ 434
  - Sort file size estimation ...................................... 435
  - RECOVER PLUS sort parameters .................................... 437
  - Allocation of sort work space ..................................... 443
- Strategies for using copies ............................................. 444
  - Image copy data set contents ..................................... 444
  - Cabinet copy data set contents .................................. 444
  - Using copies containing multiple data sets ......................... 444
  - Using data sets stacked on tape ................................... 445
  - Using image copies and log records for indexes ...................... 446
- Using R+/CHANGE ACCUM with RECOVER PLUS ............................... 448
  - Using change accumulation files for multiple objects ............... 448
  - Using separate change accumulation files for separate data sets .... 449
- Planning recovery resources ............................................. 449
  - Tips on grouping recovery resources ................................ 450
- Reading multiple log files concurrently ................................ 451
Allocating output image copy data sets dynamically ........................................ 452
Using copy data set output descriptors ......................................................... 452
GDGs and symbolic variables in data set name construction .......................... 454
Stacking copies on tape .................................................................................. 456
Using RECOVER PLUS with data sharing ..................................................... 457
Forward recovery of index spaces after altering a VARCHAR column .......... 458
Supporting real-time statistics in RECOVER PLUS ....................................... 459
Recovering altered spaces ............................................................................. 460
  Added partitions ......................................................................................... 460
  Rotated partitions .................................................................................... 461
  Changed limit keys ................................................................................... 461
Recovering encrypted copies ........................................................................ 462
  Requirements for encryption ...................................................................... 462
  Key data set .............................................................................................. 463
  Registration of copies ................................................................................ 468
Handling DB2 versioning information ............................................................. 469
Recovering cloned objects ............................................................................ 472
Recovering XML objects ................................................................................ 473
Recovering compressed indexes ................................................................... 474
Using BMC RECOVERY MANAGER groups ................................................. 476

Chapter 7          Recovering a dropped table space or table 479

Overview ........................................................................................................ 479
Example 1: Recovering a dropped table space when image copies are available ........................................................................................................ 482
Example 2: Recovering a dropped table in a segmented table space containing multiple tables .................................................................................. 487
Example 3: Recovering a dropped table from a simple table space ............ 491
Example 4: Recovering a dropped table space using a pack backup .......... 494

Chapter 8          Migrating data 497

Overview ........................................................................................................ 497
Defining target objects for migration ............................................................. 500
Example 1: Creating a consistent database for query and review ............... 501
Example 2: Moving an application to another subsystem with index recovery ............................................................................................... 508
Example 3: Moving an application to another subsystem with index rebuilding ........................................................................................... 518

Chapter 9          Making copies from the log and earlier copies 527

Overview ........................................................................................................ 527
Example 1: Creating a consistent set of copies at a point in time ............... 529
Example 2: Creating an image copy without accessing the space ............. 532

Appendix A          RECOVER PLUS installation options 535

Overview ........................................................................................................ 535
Installation options macro listing ................................................................. 536
RECOVER PLUS installation options ............................................................... 537
Appendix B  BMCSORT installation and options  561
Installation overview .......................................................... 561
DYNALOC Installation Option ................................................ 561

Appendix C  BMC utilities database  565
Overview .............................................................................. 566
  Considerations and warnings ............................................. 566
  Managing common utility tables ....................................... 567
BMCDICT table .................................................................... 569
  Considerations .................................................................. 569
  Maintaining the BMCDICT table ...................................... 570
BMCHIST table ..................................................................... 570
  Maintaining the BMCHIST table ...................................... 572
BMCLGRNX table .................................................................. 572
BMCSYNC table .................................................................... 573
  Executing BMC utilities concurrently ............................. 575
  Considerations .................................................................. 577
  Maintaining the BMCSYNC table ...................................... 577
  Cleaning up RECOVER UNLOADKEYS entries .................. 578
BMCTRANS table ................................................................... 578
BMCUTIL table ..................................................................... 580
  Maintaining the BMCUTIL table ...................................... 582
BMCXCOPY table .................................................................. 582
  Maintaining the BMCXCOPY table .................................... 586

Appendix D  BMC Common DB2 repository  587
BMC Common DB2 repository tables ..................................... 587
  Naming conventions ........................................................ 587
  Object set table ............................................................... 588
  Object set definition table ............................................... 589
  Object set SQL table ....................................................... 590
  Group options table ........................................................ 590
  Product registration table ................................................ 591
  Group authorizations table .............................................. 591

Appendix E  RECOVER PLUS syntax diagrams  593
Alphabetical listing of RECOVER PLUS options .................... 594
OPTIONS command syntax diagram .................................... 601
OUTPUT command syntax diagram ...................................... 604
RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE command syntax diagram ................................................. 606
REBUILD INDEX command syntax diagram .......................... 615
RECOVER UNLOADKEYS command syntax diagram ............... 616
RECOVER BUILDDINDEX command syntax diagram ............... 617
LOGSCAN command syntax diagram .................................... 618
# Figures

<table>
<thead>
<tr>
<th>Figure Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point-in-time recovery (forward recovery versus BACKOUT)</td>
<td>61</td>
</tr>
<tr>
<td>Unicode representation in the RECOVER PLUS output file</td>
<td>101</td>
</tr>
<tr>
<td>OPTIONS syntax diagram</td>
<td>109</td>
</tr>
<tr>
<td>OPTIONS syntax—LOGSORT specification</td>
<td>111</td>
</tr>
<tr>
<td>OPTIONS syntax—STOGROUP specification</td>
<td>111</td>
</tr>
<tr>
<td>OPTIONS syntax—SORT specification</td>
<td>111</td>
</tr>
<tr>
<td>OUTPUT syntax</td>
<td>145</td>
</tr>
<tr>
<td>OUTPUT syntax—Common options</td>
<td>145</td>
</tr>
<tr>
<td>OUTPUT syntax—Disk options</td>
<td>146</td>
</tr>
<tr>
<td>OUTPUT syntax—Tape options</td>
<td>146</td>
</tr>
<tr>
<td>RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax diagram</td>
<td>155</td>
</tr>
<tr>
<td>RECOVER syntax—Table space specification</td>
<td>156</td>
</tr>
<tr>
<td>RECOVER syntax—Single index specification</td>
<td>157</td>
</tr>
<tr>
<td>RECOVER syntax—Multiple index specification</td>
<td>158</td>
</tr>
<tr>
<td>RECOVER syntax—INDEPENDENT OUTSPACE specification</td>
<td>158</td>
</tr>
<tr>
<td>RECOVER syntax—Point-in-time recovery specification</td>
<td>159</td>
</tr>
<tr>
<td>RECOVER syntax—TOCOPY specification</td>
<td>159</td>
</tr>
<tr>
<td>RECOVER syntax—LOGSORT specification</td>
<td>159</td>
</tr>
<tr>
<td>RECOVER syntax—OBIDXLAT specification</td>
<td>160</td>
</tr>
<tr>
<td>RECOVER syntax—Non-registered copy or INLOG specification</td>
<td>160</td>
</tr>
<tr>
<td>RECOVER syntax—FROMRBA or FROMLOGPOINT specification</td>
<td>161</td>
</tr>
<tr>
<td>RECOVER syntax—OUTCOPY specification</td>
<td>161</td>
</tr>
<tr>
<td>RECOVER syntax—INCOPY specification</td>
<td>162</td>
</tr>
<tr>
<td>RECOVER syntax—OUTCOPYDDN specification</td>
<td>162</td>
</tr>
<tr>
<td>RECOVER syntax—RECOVERYDDN specification</td>
<td>163</td>
</tr>
<tr>
<td>REBUILD INDEX syntax diagram</td>
<td>213</td>
</tr>
<tr>
<td>RECOVER UNLOADKEYS syntax diagram</td>
<td>228</td>
</tr>
<tr>
<td>RECOVER BUILDINDEX syntax diagram</td>
<td>237</td>
</tr>
<tr>
<td>LOGSCAN syntax diagram</td>
<td>244</td>
</tr>
<tr>
<td>LOGSCAN syntax—Table space specification</td>
<td>245</td>
</tr>
<tr>
<td>LOGSCAN syntax—Single index specification</td>
<td>245</td>
</tr>
<tr>
<td>LOGSCAN syntax—Multiple index specification</td>
<td>246</td>
</tr>
<tr>
<td>LOGSCAN syntax—Point-in-time recovery specification</td>
<td>246</td>
</tr>
<tr>
<td>JCL for a simple RECOVER PLUS execution</td>
<td>268</td>
</tr>
<tr>
<td>RECOVER PLUS JCL for recovery of table space and index</td>
<td>298</td>
</tr>
<tr>
<td>AFRSUMRY with MSGLEVEL(0)</td>
<td>299</td>
</tr>
<tr>
<td>AFRSTMT with MSGLEVEL(0)</td>
<td>299</td>
</tr>
<tr>
<td>AFRPRINT with MSGLEVEL(0)</td>
<td>301</td>
</tr>
<tr>
<td>AFRPRINT with MSGLEVEL(1)</td>
<td>301</td>
</tr>
</tbody>
</table>
AFROSUM with MSGLEVEL(1) ........................................ 303
AFRSTMT with MSGLEVEL(2) ........................................ 304
AFRPRINT with MSGLEVEL(2) ....................................... 307
AFRPLAN with MSGLEVEL(2) ....................................... 309
Phases used for table space recovery with indexes rebuilt .......... 312
Phases used for table space recovery and parallel index sorts with  
    multitasking index rebuilds .................................... 313
Phases used for table space recovery with multiple log sorts and parallel  
    index sorts with multitasking index rebuilds .................. 315
Phases used for table space and index space recovery ............... 316
Phases used for BACKOUT recovery ................................ 317
Phases used for table space and index space recovery for Instant Snapshots 318
Example 1—JCL using ANALYZE ONLY .............................. 325
Example 1—AFRSUMRY using ANALYZE ONLY ...................... 325
Example 1—AFRSTMT using ANALYZE ONLY ....................... 326
Example 1—AFROSUM using ANALYZE ONLY ........................ 328
Example 1—AFRPLAN using ANALYZE ONLY ....................... 330
Example 1—SYSPICK using ANALYZE ONLY ......................... 333
Example 2A—JCL for a simple RECOVER TABLESPACE and  
    REBUILD INDEX ................................................. 334
Example 2A—AFRSUMRY for a simple REBUILD INDEX ............... 335
Example 2A—AFRSTMT for a simple REBUILD INDEX ................. 335
Example 2A—AFRPRINT for a simple REBUILD INDEX ............... 337
Example 2B—JCL for a simple RECOVER TABLESPACE and  
    RECOVER INDEX .................................................. 340
Example 2B—AFRSUMRY for a simple RECOVER INDEX ............... 340
Example 2B—AFRSTMT for a simple RECOVER INDEX ................. 341
Example 2B—Edited AFRPRINT for a simple RECOVER INDEX ...... 344
Edited AFROSUM for a simple RECOVER INDEX ...................... 349
Example 2B—Edited AFRPLAN for a simple RECOVER INDEX ...... 353
Example 3—JCL for recovering one partition and creating an image copy 359
Example 4A—JCL using TOCOPY with a named copy ................ 360
Example 4B—JCL using TOCOPY LASTCOPY ......................... 361
Example 5A—JCL using TOLOGPOINT with RECOVER ............... 363
Example 5B—JCL using TOLOGPOINT and BACKOUT ................. 364
Example 6—JCL recovering a partitioned table space and partitioned  
    index and unloading keys for a nonpartitioned index .......... 365
Example 7—JCL building an index with sorted unloaded keys ........ 366
Example 8A—JCL for making stacked image copies .................. 367
Example 8B—JCL for recovering with stacked cataloged image copies  
    as input ................................................................ 368
Example 9—JCL for simple table space and index recovery to non-DB2  
    data sets ................................................................ 369
Example 10A—JCL for overriding installation options .............. 372
Example 10B—JCL for overriding the IXRECP installation option 373
Example 10B—Edited AFRPRINT for overriding the IXRECP installation  
    option ................................................................. 373
Example 10B—AFRSTMT for overriding the IXRECP installation option 376
Example 10B—Edited AFRPLAN for overriding the IXRECP installation option ........................................................................................................ 379
Example 10B—AFRERR for overriding the IXRECP installation option ........................................................................................................ 382
Example 11A—JCL using R+/CHANGE ACCUM to accumulate log records ........................................................................................................ 383
Example 11B—JCL recovering a table space using a change accumulation file ..................................................................................................... 383
Example 12—JCL using the MAINT parameter .................................................................................................................................................... 384
Example 12—AFRSUMRY using the MAINT parameter ........................................................................................................................................ 384
Example 12—AFRSTMT using the MAINT parameter ......................................................................................................................................... 384
Example 13—JCL using the LOGSCAN command ............................................................................................................................................ 387
Example 13—SYSSCAN for the LOGSCAN command ....................................................................................................................................... 387
Example 14—JCL using the SIMULATE option .................................................................................................................................................... 390
Example 15—JCL using the MAXKSORT option for parallel index rebuilds ...................................................................................................... 391
Example 15—AFRSUMRY for parallel index rebuilds with MAXKSORT .......................................................................................................... 391
Example 15—AFRSTMT for parallel index rebuilds with MAXKSORT ......................................................................................................... 394
Example 15—AFRPRINT for parallel index rebuilds with MAXKSORT ......................................................................................................... 394
Example 17—JCL for table space recovery and index recovery using the MAXKSORT option ........................................................................ 397
Example 17—JCL using RECOVERYPOINT for timestamp recovery ........................................................................................................... 398
Example 18—JCL using default values ................................................................................................................................................................. 399
Example 18—AFRSTMT showing the MAXLSORT value ................................................................................................................................. 400
Example 18—SYSPICK output ............................................................................................................................................................................... 401
Example 18—AFRSUMRY ...................................................................................................................................................................................... 401
Example 18—AFRPRINT ...................................................................................................................................................................................... 402
Example 18—AFROSUM ....................................................................................................................................................................................... 404
Example 18—AFROSUM ....................................................................................................................................................................................... 405
Example 18—AFRPLAN ....................................................................................................................................................................................... 406
Multiple jobs recovering partitions of a table space ........................................................................................................................................ 412
Recovering a table space and rebuilding a partitioned index .............................................................................................................................. 413
Rebuilding a nonpartitioned index with a single job ...................................................................................................................................... 414
Rebuilding a nonpartitioned index with multiple jobs .................................................................................................................................. 415
Recovering a partitioned table space and a rebuilding nonpartitioned index .................................................................................................. 416
How PIT_RBAs and registered log points are used during LOGONLY or LOGAPPLY ONLY recoveries ................................................................. 421
When a recovery is not allowed by RECOVER PLUS .................................................................................................................................. 422
BACKOUT recovery uses the spaces and logs .................................................................................................................................................. 427
RECOVER PLUS drop recovery process ........................................................................................................................................................... 480
Example DDL for the dropped table space .................................................................................................................................................... 483
Example JCL to run DSN1PRNT ............................................................................................................................................................................. 484
Example DSN1PRNT output with the object identifiers of the dropped table space .......................................................................................... 484
Example SPUFI to locate new internal object identifiers .................................................................................................................................. 485
Example drop recovery RECOVER PLUS JCL ............................................................................................................................................... 486
Example JCL to run DSN1PRNT to obtain object identifiers for original table space ............................................................................................ 488
Example DSN1PRNT utility output with object identifiers for original table space ............................................................................................ 488
Example query for the DBID, PSID, and OBID values for the temporary table space .......................................................................................... 489
Example RECOVER PLUS drop recovery JCL .............................................................................................................................................. 490
Example DSN1LOGP utility JCL ........................................ 492
Example DSN1LOGP utility output .................................. 492
Example RECOVER PLUS JCL with dynamically allocated output copy 493
Example DDL for the dropped table space .......................... 494
Example DFDSS JCL to restore the table space data set .......... 495
Example RECOVER PLUS JCL for recovery using a pack backup 496
RECOVER PLUS migration within the same complex .......... 498
RECOVER PLUS migration to a different complex .......... 499
Creating a consistent database for query and review .......... 501
Data definitions for source objects ............................... 502
Data definitions for target objects ............................... 503
Example SQL and output to select source and target objects 503
Example QUIESCE JCL .................................................. 504
Example QUIESCE output ............................................ 505
Example RECOVER PLUS JCL to create the target image ....... 506
Query on target system .............................................. 507
Sample messages from DB2 logging .............................. 508
Example RECOVER PLUS JCL to create migration images with index recovery ........................................ 509
Data definitions for source objects ............................... 510
Example SQL and output to obtain object IDs for source objects 512
Data definitions for target objects ............................... 513
Example SQL and output to obtain object IDs for target objects 514
Example RECOVER PLUS JCL to migrate data with index recovery 516
Sample messages from DB2 logging .............................. 518
Example RECOVER PLUS JCL to create migration images ....... 519
Data definitions for the source objects ........................... 520
Example SQL and output to obtain object IDs for source objects 522
Data definitions for target objects ............................... 522
Example SQL and output to obtain object IDs for target objects 524
Example RECOVER PLUS JCL to migrate data with index rebuilding 525
Making new copies from old copies and log records ......... 528
Example JCL to QUIESCE a set of table spaces .............. 529
Example QUIESCE output ............................................ 529
Example RECOVER PLUS JCL to create copies using dynamic allocation with automatic generation of GDGs ........................................ 530
Example RECOVER PLUS JCL to create new image copy using dynamic allocation ........................................ 532
RECOVER PLUS installation options module ........................ 536
OPTIONS syntax diagram (part 1 of 2) ............................ 601
OPTIONS syntax—LOGSORT specification ......................... 603
OPTIONS syntax—STOGROUP specification ....................... 603
OPTIONS syntax—SORT specification ........................... 603
OUTPUT syntax ......................................................... 604
OUTPUT syntax—Common options .................................. 604
OUTPUT syntax—Disk options ....................................... 604
OUTPUT syntax—Tape options ...................................... 605
RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax diagram ........................................ 606
### Tables

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>System authority requirements</td>
<td>81</td>
</tr>
<tr>
<td>RECOVER PLUS actions for access to DB2 table spaces</td>
<td>84</td>
</tr>
<tr>
<td>RECOVER PLUS actions for access to DB2 index spaces</td>
<td>85</td>
</tr>
<tr>
<td>Running BMC products concurrently</td>
<td>87</td>
</tr>
<tr>
<td>RECOVER PLUS command options</td>
<td>102</td>
</tr>
<tr>
<td>Symbolic variables used with DSNAME and their descriptions</td>
<td>147</td>
</tr>
<tr>
<td>RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX</td>
<td>164</td>
</tr>
<tr>
<td>command dependencies</td>
<td></td>
</tr>
<tr>
<td>RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX</td>
<td>164</td>
</tr>
<tr>
<td>command prohibitions</td>
<td></td>
</tr>
<tr>
<td>Symbolic variables used with INDEP OUTSPACE and their descriptions</td>
<td>180</td>
</tr>
<tr>
<td>RECOVER PLUS data sets</td>
<td>277</td>
</tr>
<tr>
<td>Prefix defaults for partitioned spaces</td>
<td>281</td>
</tr>
<tr>
<td>Restarts allowed for each phase of RECOVER PLUS</td>
<td>292</td>
</tr>
<tr>
<td>Major RECOVER PLUS execution phases</td>
<td>311</td>
</tr>
<tr>
<td>Cross reference of examples by command or keyword</td>
<td>320</td>
</tr>
<tr>
<td>Parameters for DASD Work Space</td>
<td>442</td>
</tr>
<tr>
<td>Symbolic variables for specifying data set names</td>
<td>455</td>
</tr>
<tr>
<td>Columns updated by the RECOVER PLUS RECOVER command for real-time</td>
<td>459</td>
</tr>
<tr>
<td>statistics</td>
<td></td>
</tr>
<tr>
<td>Columns updated by the RECOVER PLUS REBUILD command for real-time</td>
<td>460</td>
</tr>
<tr>
<td>statistics</td>
<td></td>
</tr>
<tr>
<td>RECOVER PLUS installation options</td>
<td>537</td>
</tr>
<tr>
<td>DYNALOC parameters</td>
<td>563</td>
</tr>
<tr>
<td>Common utility tables</td>
<td>566</td>
</tr>
<tr>
<td>BMCDICT table</td>
<td>569</td>
</tr>
<tr>
<td>BMCHIST table</td>
<td>571</td>
</tr>
<tr>
<td>BMCLGRNX table</td>
<td>572</td>
</tr>
<tr>
<td>BMCSYNC table</td>
<td>573</td>
</tr>
<tr>
<td>Executing BMC utilities concurrently</td>
<td>575</td>
</tr>
<tr>
<td>BMTRANS table</td>
<td>578</td>
</tr>
<tr>
<td>BMCUTIL table</td>
<td>580</td>
</tr>
<tr>
<td>BMXCOPY table</td>
<td>583</td>
</tr>
<tr>
<td>BMC Common DB2 repository synonym and local table names</td>
<td>588</td>
</tr>
<tr>
<td>OBJSETS table</td>
<td>588</td>
</tr>
<tr>
<td>OBJSET_DEF table</td>
<td>589</td>
</tr>
<tr>
<td>OBJSET_SQL table</td>
<td>590</td>
</tr>
<tr>
<td>GRPOPTS table</td>
<td>591</td>
</tr>
<tr>
<td>PRODREG table</td>
<td>591</td>
</tr>
<tr>
<td>GROUPAUTH table</td>
<td>591</td>
</tr>
</tbody>
</table>
About this book

This book contains detailed information about the RECOVER PLUS for DB2® product and is intended for DB2 system administrators and DB2 database administrators (DBAs).

To use this book, you should be familiar with the following items:

- IBM® DB2 Universal Database for z/OS® (DB2) DBMS
- z/OS operating system
- job control language (JCL)
- Interactive System Productivity Facility (ISPF)

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 Customer Support website at http://www.bmc.com/support_home. Most product shipments also include the online books on a documentation CD.

**NOTE**

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

The following related publications supplement this book.

<table>
<thead>
<tr>
<th>Category</th>
<th>Document</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>installation documentation</td>
<td>Backup and Recovery Products for DB2 Installation Guide or Recovery Management for DB2 Installation Guide</td>
<td>provides information about installing and customizing RECOVER PLUS and other BMC Backup and Recovery products for DB2</td>
</tr>
<tr>
<td></td>
<td>BMC Products and Solutions for DB2 for z/OS Installation Planning Guide</td>
<td>contains information about BMC products and solutions for DB2, helping you understand the relationship between the products and solutions and plan their installation in your environment</td>
</tr>
<tr>
<td></td>
<td>online Help panels for RECOVER PLUS options on the Installation System panels</td>
<td>provides information for fields that are required when installing the RECOVER PLUS product</td>
</tr>
<tr>
<td>core documents</td>
<td>Backup and Recovery Products for DB2 Messages Manual</td>
<td>provides message and return code information for all the BMC Backup and Recovery products for DB2 and any components used by the products</td>
</tr>
<tr>
<td>documents for associated products and technology components</td>
<td>COPY PLUS for DB2 Reference Manual</td>
<td>provides information about using COPY PLUS</td>
</tr>
<tr>
<td></td>
<td>EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide</td>
<td>provides instructions for using the I/O caching and snapshot processing features of the BMC EXTENDED BUFFER MANAGER (XBM) product and its associated SNAPSHOT UPGRADE FEATURE (SUF)</td>
</tr>
<tr>
<td></td>
<td>R+/CHANGE ACCUM for DB2 User Guide</td>
<td>describes how to use the R+/CHANGE ACCUM product</td>
</tr>
<tr>
<td></td>
<td>Recovery Management for DB2 User Guide</td>
<td>provides information about the BMC Recovery Management for DB2 solution, of which RECOVER PLUS is a component</td>
</tr>
<tr>
<td></td>
<td>RECOVERY MANAGER for DB2 User Guide</td>
<td>provides information about using RECOVERY MANAGER</td>
</tr>
<tr>
<td>notices</td>
<td>release notes, flashes, technical bulletins</td>
<td>explain the latest updates to RECOVER PLUS</td>
</tr>
</tbody>
</table>
Subsequent chapters also refer to these IBM books:

- DB2 for z/OS Administration Guide
- DB2 for z/OS Codes
- DB2 for z/OS Data Sharing: Planning and Administration
- DB2 for z/OS Messages
- DB2 for z/OS SQL Reference

**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

- Change bars show where substantive technical changes have been made to the document since its previous publication. These changes include clarifications or corrections to existing information, and new information that corresponds to product changes.

- This book uses the following types of special text:

  __NOTE__
  Notes contain important information that you should consider.

  __WARNING__
  Warnings alert you to situations that could cause problems, such as loss of data, if you do not follow instructions carefully.

  __TIP__
  Tips contain information that might improve product performance or that might make procedures easier to follow.
Syntax diagrams

The following figure shows the standard format for syntax diagrams:

The following example illustrates the syntax for DELETE. Because the FROM keyword, alias variable, and WHERE clause are optional, they appear below the main command line. In contrast, the tableName variable appears on the command line because the table name is required. If the statement includes a WHERE clause, the clause must contain a search condition or a CURRENT OF clause. (The searchCondition variable appears on the main line for the WHERE clause, indicating that this choice is required.)
The following guidelines provide additional information about syntax diagrams:

- Read diagrams from left to right and from top to bottom.

- A recursive (left-pointing) arrow above a stack indicates that you may choose more than one item in the stack.

- An underlined item is a default value.

- If a diagram shows punctuation marks, parentheses, or similar symbols, you must enter them as part of the syntax. Asterisks are exceptions. An asterisk in a diagram indicates a reference note.

- In general, operating system commands, keywords, clauses, and data types are displayed in uppercase letters. However, if an item can be shortened, the minimum portion of the operating system command or keyword might be displayed in uppercase letters with the remainder of the word in lowercase letters (for example, CANcel).

- The following conventions apply to variables in syntax diagrams:

  — Variables typically are displayed in lowercase letters and are always italicized.
  — If a variable is represented by two or more words, initial capitals distinguish the second and subsequent words (for example, databaseName).

- A part of the syntax diagram that is boxed and normally followed by the word “specification” indicates that the portion of the syntax is too large or complex to include in the statement syntax diagram overview. A page reference below the boxed specification gives the location of the figure showing the additional syntax.
Summary of changes

This section summarizes changes to the functionality of the product, listing the changes by product version and release date. The summary includes enhancements to the product and any major changes to the documentation.

Version 10.1.00  April 2011

This release of RECOVER PLUS includes the following product enhancements and changes:

**DB2 Version 10 support**

RECOVER PLUS supports the following DB2 Version 10 features and changes:

- auto-compression (compress on INSERT)
- catalog changes and restructuring
- DEFINE NO LOB and XML spaces
- new DBA authorities system DBADM and DATAACCESS

RECOVER PLUS now checks for these two new authorities prior to execution.

System DBADM and DATAACCESS are added to the list of accepted authorities (page 80).

- IBM FlashCopy® image copies

RECOVER PLUS can use IBM FlashCopy image copies as a recovery resource (similar to BMC Snapshot copies). RECOVER PLUS processes FlashCopy image copies during the RECOVER PLUS SNAP phase:

- processes a consistent FlashCopy in the same way as a BMC Online Consistent Copy

- processes an inconsistent FlashCopy as a SHRLEVEL CHANGE copy (in the same way as inconsistent BMC Snapshot copies)

- processes sequential copies created from a consistent FlashCopy in the same way as sequential BMC Online Consistent Copies

- processes sequential copies created from an inconsistent FlashCopy as SHRLEVEL CHANGE copies
The OPTION RESOURCE SELECTION FC syntax (page 120) supports FlashCopy. FlashCopy is also supported in the installation options LOCCPSEL (page 551) and REMCPSEL (page 551) by using the FC parameter. The default processing order for local copies is FC, LP, LB. The default processing order for remote copies is RP, RB, FC.

This release adds the FLASHCOPY option to the non-registered copy or INLOG specification with INCOPY FULL (page 195). Also, updates to the DSNAME description of the INCOPY specification (page 204) accommodate FlashCopy support.

- hash access to data

RECOVER PLUS recovers table spaces with hashing access enabled and supports new log records and page types that are defined for hashing. When RECOVER PLUS recovers a hashed table space to a point-in-time prior to the point when hashing was enabled, the table space is placed in advisory REORG-pending (AREOR) status; the hash overflow index is placed in REBUILD-pending (RBDP) status. RECOVER PLUS also supports a REBUILD of the hash overflow index.

- include columns or additional non-key columns in unique indexes

- inline LOBs

- segmented MEMBER CLUSTER for universal table spaces (UTSs)

- pending definition changes (pending ALTERs)

  RECOVER PLUS supports the new SYSIBM.SYSCOPY entries that this feature generates.

- 64-bit runtime

- skip-level migration

RECOVER PLUS supports migrating to DB2 Version 10 from DB2 Version 8, which introduces several new migration modes:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CM8</td>
<td>conversion mode from DB2 Version 8</td>
</tr>
<tr>
<td>CM8*</td>
<td>conversion mode* from DB2 Version 8</td>
</tr>
<tr>
<td>CM9</td>
<td>conversion mode from DB2 Version 9</td>
</tr>
<tr>
<td>CM9*</td>
<td>conversion mode* from DB2 Version 9</td>
</tr>
<tr>
<td>ENFM8a</td>
<td>enabling-new-function mode from DB2 Version 8</td>
</tr>
<tr>
<td>ENFM8*</td>
<td>enabling-new-function mode* from DB2 Version 8</td>
</tr>
<tr>
<td>ENFM9a</td>
<td>enabling-new-function mode from DB2 Version 9</td>
</tr>
</tbody>
</table>
Summary of changes

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENFM9*</td>
<td>enabling-new-function mode* from DB2 Version 9</td>
</tr>
<tr>
<td>NFM</td>
<td>new-function mode</td>
</tr>
</tbody>
</table>

Before using BMC products in this mode, you must run the IBM job DSNTIJEN to successful completion. DSNTIJEN converts DB2 to enabling-new-function mode from DB2 Version 8 or 9.1. Successful completion of DSNTIJEN completes catalog migration. BMC does not support DB2 catalogs that are not completely migrated.

- greater timestamp precision (extends microseconds to 12 places, but 6 remains the default)
- TIMESTAMP WITH TIME ZONE data type
- archive logs and sequential image copy data sets in the cylinder-managed portion of Extended Address Volumes (EAVs)

RECOVER PLUS can process archive logs and sequential image copy data sets that have been written to the cylinder-managed portion of EAVs.

RECOVER PLUS can also write output image copies with the EAV attribute. To support this capability, EATTR has been added to the OUTPUT command (page 151), and EATTR, STORCLAS and DATACLAS (page 557 and page 559) have been added to the installation options.

**NOTE**

For IBM z/OS versions earlier than 1.11, you must set the EATTR option to NONE.

If an image copy was written to the cylinder-managed portion of an EAV under z/OS Version 1.11, you cannot use that image copy on z/OS Version 1.10; Version 1.10 does not support sequential data sets in the cylinder-managed portion of an EAV.

- last/currently committed data
  
  RECOVER PLUS supports the new log records that this feature generates.

- temporal tables

To support temporal tables, RECOVER PLUS adds the HISTORY keyword to the AUX installation option (page 555). Using this keyword includes history spaces when a table space with system-maintained temporal tables is specified in SYSIN. You can also specify AUX HISTORY at runtime (and override the installation option value) on the following RECOVER PLUS commands:

— OPTIONS (page 122)
— RECOVER TABLESPACE (page 169)
— RECOVER INDEX(ALL) TABLESPACE (page 175)
Summary of changes

— REBUILD INDEX(ALL) TABLESPACE (page 219)
— LOGSCAN TABLESPACE (page 250)
— LOGSCAN INDEX(ALL) TABLESPACE (page 254)

Similar to DB2 RECOVER Version 10 VERIFYSET NO behavior, RECOVER PLUS does not check

— that the system-maintained temporal space and the history space are recovered as a set

— that the recovery of the system-maintained temporal space and the history space is to the same log point

— that all DSNUMs are recovered in the same SYSIN

- XML multi-versioning

- XML indexes that are created with DATE and TIMESTAMP data

RECOVER PLUS supports recovering and rebuilding of XML user-defined indexes that are created with the AS SQL DATE and AS SQL TIMESTAMP options.

XBM zIIP redirection support

RECOVER PLUS now provides the option to offload eligible processing to an IBM System z® Integrated Information Processor (zIIP). To enable and use zIIP processing, you must have an installed and authorized version of the EXTENDED BUFFER MANAGER (XBM) product or the SNAPSHOT UPGRADE FEATURE (SUF) technology.

The new ZIIP command and installation options enable this functionality. For more information, see “ZIIP” on page 130 or page 547.

You can also use the existing XBMID installation or command option to specify an XBM subsystem through which to access this functionality. For more information, see “XBMID ssid or xbmGroup” on page 129 or page 547.

For more information about the XBM component that enables the use of zIIPs, see the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide.

Versioned, compressed indexes support

RECOVER PLUS now supports versioned, compressed indexes. For versioned, compressed indexes, you must recover all of the data sets in the same step.
**COMPRESSED option**

RECOVER PLUS adds the COMPRESSED option to the INCOPY clause of the RECOVER command (page 162, page 207). The COMPRESSED option allows you to indicate whether image copies of compressed indexes are compressed copies.

**Indexes on expression**

RECOVER PLUS provides native support for some indexes on expression with REBUILD INDEX.

**Log sort estimates**

RECOVER PLUS improves log sort estimates for multitasking.

**OBIDXLAT improvements**

RECOVER PLUS has improved OBIDXLAT processing for recovering one data set of a multi-data-set, nonpartitioned space.

RECOVER PLUS now translates to the target OBID, regardless of the source OBID, when the following conditions:

- The header page is not available.
- The object is an index or a single-table table space.
- No OBIDs are specified.

In earlier versions, OBIDXLAT processing failed.

Also, RECOVER PLUS no longer attempts to read the header pages from the 2nd - n data sets of multi-data-set, nonpartitioned spaces.

**OBIDXLAT with OBJECTSET**

RECOVER PLUS now allows OBIDXLAT with OBJECTSET.

**USEHDROBIDS option**

RECOVER PLUS adds the USEHDROBIDS option on the OPTIONS command (page 123) and as an installation option (page 557). USEHDROBIDS indicates whether to use the OBIDs in the header page for MERGE phase optimization and checking (comparing the source image copy header page OBIDs to the data page OBIDs). The default value is YES.
NOCOPYPEND option

RECOVER PLUS adds the NOCOPYPEND option to REBUILD INDEX (page 227) to reset the ICOPY status on COPY YES indexes.

DSNUM begin:end and PART begin:end

RECOVER PLUS removes the requirement for spaces on either side of the colon for DSNUM begin:end (page 169, page 173, page 250, and page 253) and PART begin:end (page 215, page 221, page 230, and page 231) syntax.

NUMREC changes for key sorts

For key sorts only, RECOVER PLUS has changed some NUMREC parameters for REBUILD INDEX (page 224) and RECOVER UNLOADKEYS (page 234):

- For NUMREC CALC, RECOVER PLUS uses a minimum estimate of 400,000 for each index key sort (or 200,000 if the key length is greater than 1000 characters).

- For NUMREC NOEST, RECOVER PLUS passes an estimate of 400,000 to each index key sort (or 200,000 if the key length is greater than 1000 characters).

- For NUMREC EST integer, if the specified estimate is less than 400,000 (200,000 for keys with a length greater than 1000 characters), RECOVER PLUS uses an estimate of 400,000 (or 200,000 if the key length is greater than 1000 characters).

Real-time statistics for index keys

RECOVER PLUS uses real-time statistics (RTS) to estimate the number of keys for each index (page 288). When the following conditions are met, RECOVER PLUS uses the TOTALROWS value from table SYSIBM.SYSTABLESPACESTATS for single-table table spaces:

- You are using DB2 Version 9 or later.
- UTSORTAL is set to YES.

**NOTE**

If you perform a migration, TOTALROWS might not be accurate. If it is not accurate after the migration, you must reset TOTALROWS to NULL. However, resetting TOTALROWS is usually unnecessary in the following situations:

- if you migrate data daily and the size of the space does not change significantly
- if you include the real-time statistics tables in your migration
Summary of changes

**OBJECTSET syntax diagram correction**

OBJECTSET syntax diagrams throughout this book have been corrected by removing the comma and recursive arrow that indicated that multiple OBJECTSET `creator.name` specifications were valid in a single clause.

Only one OBJECTSET `creator.name` specification is valid in a single clause.

**Version 9.2.00 Revised—May 2010**

The *RECOVER PLUS for DB2 Reference Manual* is revised to correct an error in the December 2009 edition. All change bars from the December 2009 edition are retained.

The MODEL syntax diagram in Figure 23 on page 162 and description on page 205 were incorrect. This book now shows the correct diagram and description:

```
**NOTE**

When you use INCOPY with TRANSFORM, specify the VSAM data component instead of the VSAM cluster.
```
Version 9.2.00  December 2009

This update of RECOVER PLUS includes the following product enhancements and changes:

**AUX option**

RECOVER PLUS adds the AUX option to specify if auxiliary objects will be included with the recovery of the base table spaces for XML and LOB spaces. Valid values are NO, YES, XML, and LOB. The default value is NO (on the installation option).

The AUX option is available on the OPTIONS command (page 122) and as an installation option (page 555). Additionally, you can specify AUX with the table space specification to determine if the auxiliary table space or indexes will be included in the following RECOVER PLUS statements:

- RECOVER TABLESPACE
- RECOVER INDEX(ALL) TABLESPACE
- REBUILD INDEX(ALL) TABLESPACE
- LOGSCAN TABLESPACE
- LOGSCAN INDEX(ALL) TABLESPACE

**NOTE**

RECOVER PLUS ignores the AUX option when you specify OBJECTSET.

Also, the AUX option has no effect for the ACCUM TABLESPACE, RECOVER UNLOADKEYS, and RECOVER BUILDINDEX commands.

You cannot specify the following options with AUX YES, AUX XML, or AUX LOB:

- TOCOPY **dataSetName**
- OBIDXLAT without defaults
- OUTCOPYDSN
- RECOVERYDSN

**NOCOPYPEND option**

RECOVER PLUS adds the NOCOPYPEND option (page 197) to the non-registered copy or INLOG specification (page 160) of the RECOVER command. You can use NOCOPYPEND to reset the COPY-pending status when you use INCOPY.
Summary of changes

Dynamic grouping, the OBJECTSET option, and the INDEXES option

RECOVER PLUS now uses dynamic grouping for RECOVERY MANAGER groups and reads the new BMC Common DB2 repository. (page 476 and page 587) Dynamic grouping resolves the table space and index object names for inclusion with the various RECOVER PLUS commands that support OBJECTSET object types.

NOTE

Because of the new repository used by RECOVERY MANAGER, be aware of the following considerations:

- RECOVER PLUS versions earlier than version 9.2.00 will not be compatible with RECOVERY MANAGER version 9.2.00.
- RECOVER PLUS version 9.2.00 will not be compatible with RECOVERY MANAGER versions earlier than version 9.2.00.

RECOVER PLUS adds the OBJECTSET syntax to work with groups, which are defined using the RECOVERY MANAGER for DB2 product. You can use OBJECTSET with the following RECOVER PLUS commands:

- RECOVER
- SIMRCVR
- REBUILD INDEX
- SIMBLD INDEX
- LOGSCAN

See the syntax descriptions for each command in Chapter 3, “RECOVER PLUS syntax.”

The INDEXES option, with a synonym of INDEX, has been added to the RECOVER and LOGSCAN syntax in order to be able to pick up indexes that are not included in a group but that are associated with the specified table space. (page 168, page 249)

The long range plan is that OBJECTSET will be used as common syntax by other BMC utilities for DB2. Following are some examples of the use of the OBJECTSET syntax:

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Objects copied</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER OBJECTSET <code>creator.name</code></td>
<td>recovers all objects in the object set—table spaces and indexes</td>
</tr>
<tr>
<td></td>
<td>Note that INDEXES YES is not valid with this specification.</td>
</tr>
<tr>
<td>RECOVER TABLESPACE OBJECTSET <code>creator.name</code></td>
<td>recovers only table spaces in the object set</td>
</tr>
</tbody>
</table>
**Summary of changes**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Objects copied</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER TABLESPACE OBJECTSET creator.name INDEX YES&lt;sup&gt;a&lt;/sup&gt;</td>
<td>recovers table spaces in the object set along with their associated indexes, regardless of whether the indexes are included in the group</td>
</tr>
<tr>
<td>RECOVER INDEX OBJECTSET creator.name&lt;sup&gt;a&lt;/sup&gt;</td>
<td>recovers only indexes in the object set</td>
</tr>
<tr>
<td>REBUILD INDEX OBJECTSET creator.name&lt;sup&gt;b&lt;/sup&gt;</td>
<td>rebuilds all indexes in the object set</td>
</tr>
<tr>
<td>LOGSCAN OBJECTSET creator.name</td>
<td>log scan processing of all objects in the object set—table spaces and indexes</td>
</tr>
<tr>
<td>LOGSCAN TABLESPACE OBJECTSET creator.name</td>
<td>log scan processing for only table spaces in the object set</td>
</tr>
<tr>
<td>LOGSCAN TABLESPACE OBJECTSET creator.name INDEX YES</td>
<td>log scan processing for table spaces in the object set along with their associated indexes</td>
</tr>
<tr>
<td>LOGSCAN INDEX OBJECTSET creator.name</td>
<td>log scan processing for only indexes in the object set</td>
</tr>
</tbody>
</table>

<sup>a</sup> SIMRCVR is also valid.

<sup>b</sup> SIMRBLD is also valid.

**RESETRTS option**

RECOVER PLUS adds the RESETRTS option (page 203) to the OUTCOPY specification (page 161) of the RECOVER command. You can use RESETRTS to have RECOVER PLUS reset the copy DB2 real-time statistics (RTS) if the copy is registered when you specify OUTCOPY YES or OUTCOPY ONLY.

You should specify RESETRTS only if the OUTCOPY you are making is a copy that approximately represents the current state of the space. If you specify RESETRTS for a copy corresponding to a different point in time, the RTS may give misleading information about the time of the last copy and the changes made since the copy.

**Relative generation numbers for TOCOPY LASTCOPY and TOLOGPOINT LASTQUIESCE**

RECOVER PLUS adds syntax to support relative generation numbers for RECOVER...TOCOPY LASTCOPY (page 186) and RECOVER...TOLOGPOINT LASTQUIESCE (page 182).

**Generation numbers for LOGMARK**

RECOVER PLUS adds syntax to support relative and absolute generation numbers for a log mark on the OPTIONS command (page 143) and for RECOVER...TOLOGPOINT LOGMARK (page 183).
Summary of changes

**TZRULE option for daylight savings time adjustment**

The TZRULE installation option replaces the FALLTS, PFALLTS, and SPRINGTS installation options. TZRULE enables you to specify the rules for when DST begins and ends. Using TZRULE, you do not need to update the value yearly, as was the case with the previous options. (page 548)

**Speed up OBIDXLAT**

OBIDXLAT for Instant Snapshots of universal table spaces (UTS) is now much faster, even if the OBIDs change. No syntax change is required to take advantage of this enhancement. If you specify RESET, RECOVER PLUS does not get a speed improvement. However, for data sharing environments, RESET is usually not necessary. For non-data-sharing environments, if the RBAs on the target system are larger than the RBAs on the source, RESET is probably not necessary. (page 194)

**EAV support**

RECOVER PLUS supports Extended Address Volumes (EAVs) for DB2 spaces, Instant Snapshot copies, active logs, and the BSDS. EAV support requires z/OS Version 1.10 and later. (page 78)

**SORTNUM changes**

The maximum value for SORTNUM changed from 99 to 255. Also, if you do not specify a value for the SORTNUM installation option, the value is obtained from the BMCSORT installation options.

**ADFCOPY and ACA1LOGP removed**

The documentation no longer references ADFCOPY and ACA1LOGP.

**SORTDSM and SORTPERF removed**

The documentation no longer references SORTDSM and SORTPERF. These parameters were used with SYNCSORT, and RECOVER PLUS no longer uses SYNCSORT for sorting. All references to Dynamic Storage Management have been removed.

**AFRINFLT removed**

References to the AFRINFLT data set have been removed from this book because there is no reason to define this data set for RECOVER PLUS. (AFRINFLT is used only with Online Consistent Copies. See the Recovery Management for DB2 User Guide for information.)
BMCSORT installation options documentation

Appendix B, “BMCSORT installation and options” has been added to provide information about BMCSORT. A separate document for BMCSORT is no longer produced.

Recovery Management support

RECOVER PLUS adds the following enhancements to support the Recovery Management solution. You must use a Recovery Management solution password for these enhancements.

- Adds the BACKOUT installation option with AUTO as the default value for point-in-time recoveries (page 556): If you are using a Recovery Management password and you do not specify BACKOUT on the OPTIONS statement, a point-in-time recovery will default to BACKOUT AUTO. If you are not using a Recovery Management password, a point-in-time recovery is a standard forward recovery.

- In conjunction with the addition of the BACKOUT installation option, the BACKOUT option on the OPTIONS command adds syntax for BACKOUT NO and BACKOUT YES. (page 124)

- High-speed Structure Change process: Recovery Management allows the following changes to DB2 structures and data with little or no outage (see the Recovery Management for DB2 User Guide for details):
  - simple table space to partitioned by growth (PBG)
  - non-large partitioned table space to large partitioned table space
  - partitioned by range (PBR) table space to universal table space (UTS) PBR
  - PBR table space to UTS PBG

At a more granular level, you can use the High-speed Structure Change process to simply change the segment size of a table space. This includes transforming a non-segmented table space into a segmented one. You can also change the data set size of your table space or index. This is helpful if you are reaching a capacity limit, for example, exceeding the data set size or the number of data sets.

RECOVER PLUS performs the page transformations for this process. The TRANSFORM option (page 171), the DSSIZE option (page 205), and the PIECESIZE option (page 205) for High-speed Structure Change have been added to this book.
Summary of changes

Version 9.1.00 - Revised April 2009

NOTE
This revision of the RECOVER PLUS Reference Manual retains change bars from the November 2008 release and includes change bars for the new enhancements noted in this section for April 2009.

This update of RECOVER PLUS includes the following product enhancements and changes:

Parallel MERGE phases and multiple log sorts

RECOVER PLUS supports multiple log sorts and parallel MERGE phases to improve performance. For more information, see “Setting the MAXLSORT and KSORTSHARE options” on page 441. In order to provide this support the following additions and changes have been made:

- added the MAXLSORT installation option (page 553) and also added MAXLSORT to the OPTIONS command (page 138)

MAXLSORT specifies the maximum number of log sorts that can run concurrently and also determines the number of MERGE/RESTORE/SNAP phases that can run in parallel, whether or not log records are processed. The MAXLSORT installation option default is 0, which indicates that RECOVER PLUS will set an appropriate value.

- added the KSORTSHARE installation option (page 554), which defaults to YES, and also added KSORTSHARE to the OPTIONS command (page 139)

KSORTSHARE specifies if key sorts are shared among RECOVER PLUS table space recoveries (MERGE phases) running in parallel.

- changed DD names for log sort work data sets
  - changed LOGSWKxx to LxxxWKnn
  
  In the new naming convention, xxx represents the number of the log sort and nn represents the number of the work data set.

  - changed the SYSOUT data set for messages from log sorts to LOGOUxxx

  In LOGOUxxx, xxx represents the number of the log sort.
- changed message order

When MAXLSORT is greater than 1, the MERGE/SNAP/RESTORE phases run in subtasks in parallel. The order of execution of phases may be different from the execution plan in AFRPLAN. RECOVER PLUS uses deferred messaging for these phases and phase messages are printed in AFRPRINT in the order in which the phases complete rather than in the order specified in AFRPLAN. The order of message output for phases may not be repeatable from job to job. (As in previous releases, AFRSUMRY messages are printed in execution plan order.)

**Version 9.1.00  November 2008**

This release of RECOVER PLUS includes the following product enhancements and changes:

**Native support for compressed indexes**

RECOVER PLUS now natively supports the recovery of compressed indexes. For more information, see “Recovering compressed indexes” on page 474.

**Support for multi-volume disk output for cabinet copies**

RECOVER PLUS version 9.1.00 removes the version 8.1.00 restriction that did not support multi-volume disk output for cabinet copies (page 374).

**BMCHIST table support**

RECOVER PLUS now supports updates to the BMCHIST table (page 570). Additionally, RECOVER PLUS now includes the HISTORY installation option, which defaults to YES to update the BMCHIST table (page 553).

**LOGMARK addition**

RECOVER PLUS now supports using a log mark from the BMC Log Master product as a recovery point. The following new syntax for the RECOVER command allows a point-in-time recovery specification with a log mark (page 183):

```
TOLOGPOINT LOGMARK logMarkName
```

The following new syntax for the OPTIONS command allows a timestamp recovery to use a log mark (page 143):

```
OPTIONS RECOVERYPOINT LOGPOINT LOGMARK logMarkName
```

The RECOVERYPOINT syntax requires that you have a Recovery Management password.
Summary of changes

&DSNUM, &PART, &LSDUM, and &LPART symbolic variables

This release changed the following symbolic variables, implementing them in the same way that they are implemented in COPY PLUS:

- &DSNUM or &PART data set or partition (2 digits for parts 0 - 99, 3 digits for parts 100 - 999, 4 digits for parts greater than 999)
- &LDSNUM or &LPART data set or partition (3 digits for parts 0 - 999, 4 digits for parts greater than 999)

This change also includes a change to the text of message BMC96179:

THE INCOPY MODEL DSNAME MUST CONTAIN THE SYMBOLIC &DSNUM OR &LDSNUM

OBID translation performance improvement

When you use Instant Snapshots for migration and specify the OBIDXLAT option, RECOVER PLUS automatically detects whether the OBIDs have changed. If the OBIDs have not changes, RECOVER PLUS avoids the overhead of reading and writing every page. (page 194)

Support for migration of versioned spaces

This release adds the new syntax UPDATE VERSIONS to the RECOVER command. This option tells RECOVER PLUS to call the DB2 REPAIR VERSIONS utility after completing a table space recovery of a versioned table space. The DB2 REPAIR VERSIONS utility updates the version information in the DB2 catalog and directory so that the source and target information match. UPDATE VERSIONS is useful for migrations in which RECOVER PLUS restores a copy from a source system to a target system. (page 192)

Support for non-data-sharing, striped files

This release removes the limitation on data striping of active log data sets. When you apply IBM APAR PK64576 to DB2, RECOVER PLUS now reads striped active logs in non-data-sharing DB2 subsystems. (See “RECOVER PLUS limitations” on page 55.)
Summary of changes

Default value for the MAXLOGS installation option

To improve RECOVER PLUS processing, the default value of the MAXLOGS installation option (page 545) has changed from 3 to 5.

Future removal of ADFCOPY and ACA1LOGP

Future releases of RECOVER PLUS will not include the ADFCOPY and ACA1LOGP diagnostic utilities. This release issues the following messages as notification:

| BMC26346W DIAGNOSTIC UTILITY ACA1LOGP WILL NOT BE SHIPPED IN FUTURE RELEASES |
| BMC96172W DIAGNOSTIC UTILITY ADFCOPY WILL NOT BE SHIPPED IN FUTURE RELEASES |

Execution of these utilities results in RC = 4 or greater.

BMCXCOPY table additions

This release adds the CAB_BLOCK column to the BMCXCOPY table for cabinet copy support. Additionally, the NOTE_TYPE column of the BMCXCOPY has an additional value of F for frame. (page 582)

Changes LOGSWKnn to L001WKnn

The sort work file naming convention has changed from LOGSWKnn to L001WKnn in preparation for a future enhancement for multiple, parallel log sorts.

Uses DSNHDECP module value for the EXEC statement ssid value

If you do not code the DB2 subsystem ID (ssid) in the runtime parameters on the JCL EXEC statement, RECOVER PLUS uses the default value from the DSNHDECP module that is in the STEPLIB concatenation. See “DB2 subsystem identifier (ssid)” on page 271.

Removal of support for DB2 Version 7

Summary of changes

**Documentation changes**

This release includes the following documentation changes:

- **Removal of DUPOKS and MAXDUPS**—All references to the DUPOKS and MAXDUPS installation options have been removed. (RECOVER PLUS version 8.1.00 included message BMC96169I to alert you of this change.)

- **Combines the User Guide and Reference Manual**—The *RECOVER PLUS Reference Manual* now includes the information that was in the *RECOVER PLUS User Guide*. The reference manual has reordered the initial chapters to match the order used in other BMC utilities for DB2 reference manuals.
# Introducing RECOVER PLUS

This chapter explains what RECOVER PLUS does and how it works.

Overview ................................................. 44
Solution integration ..................................... 45
Features and benefits of RECOVER PLUS ............... 46
  Reduced elapsed time required to recover .......... 46
  Better audit and recovery process control .......... 48
  Expanded recovery capabilities ................. 48
  Recovery test capability .......................... 50
  TRANSFORM and High-speed Structure Change ..... 50
RECOVER PLUS operation ............................... 51
  Using IBM DB2 resources ........................... 53
  Using RECOVER PLUS with RECOVERY MANAGER .. 54
  Using RECOVER PLUS with R+/CHANGE ACCUM ..... 54
RECOVER PLUS limitations .............................. 55
Planning recovery ...................................... 57
Recovery strategies ..................................... 58
  Using LOGSCAN for estimates ....................... 58
  Using BACKOUT for point-in-time recovery without image copies 59
  Comparing BACKOUT performance .............. 60
  Using BACKOUT AUTO recovery ................. 61
  Using copies and logs for index recovery ....... 62
  Performing forward recovery without an image copy . 62
  Recovering with Instant Snapshot copies ........ 62
  Using parallel MERGE phases and log sorts ....... 63
  Using parallel MERGE phases and log sorts ...... 63
  Eliminating interim data sets for index keys .... 64
  Rebuilding nonpartitioned indexes on a partitioned table space 64
  Using resource selection ......................... 64
  Previewing recovery activity ..................... 65
  Testing recovery with simulation ................. 66
  Recovering while activity continues .......... 68
  BMCXCOPY registrations .......................... 68
  Creating copies from recovery resources .......... 69
  Overriding registered copies ..................... 69
  Recovering LOB spaces with RECOVER PLUS ....... 69
Overview

The BMC RECOVER PLUS for DB2 product provides an alternative to the IBM DB2 RECOVER utility in many functional areas. RECOVER PLUS is a batch utility that runs outside the DB2 subsystem and provides faster execution through advanced I/O techniques and the use of alternate recovery strategies. RECOVER PLUS includes the following features:

- ability to create an image copy in parallel with the recovery
- ability to create SHRLEVEL REFERENCE full copies without any impact to the real spaces (OUTCOPY ONLY option)
- dynamic allocation of the output image copies
- BACKOUT recovery
- recovery that uses Instant Snapshot copies made by the BMC COPY PLUS product
- automatic fallback
- recovery simulation, when you use the Recovery Management for DB2 solution
- consistent recovery to any point in time without a pre-established quiesce point, when you use the Recovery Management for DB2 solution
- the ability to recover a large number of data sets with a single data set allocation and deallocation when you use a cabinet copy, which you can create if you have a Recovery Management solution password
RECOVER PLUS also supports

- data sharing
- LARGE table spaces
- inline copies
- ASCII data
- UNICODE data
- LOB table spaces
- XML table spaces and index spaces
- universal table spaces
- clone spaces
- encrypted copies

RECOVER PLUS offers enhanced concurrency when accessing DB2 resources and provides additional options to improve processing efficiency. The ability of RECOVER PLUS to analyze and report planned recovery activity provides a high degree of confidence in the predictability of the recovery process.

Because of these improvements in RECOVER PLUS, you can meet more aggressive customer demands for service-level agreements on down time than does DB2 RECOVER.

Solution integration

RECOVER PLUS is a component of the Recovery Management for DB2 solution. This solution integrates the features of the following BMC products and technologies:

- RECOVERY MANAGER for DB2
- RECOVER PLUS for DB2
- R+/CHANGE ACCUM
- COPY PLUS for DB2
- Log Master for DB2 with High-speed Apply Engine
- SNAPSHOT UPGRADE FEATURE (SUF), which is a licensed component of the EXTENDED BUFFER MANAGER (XBM) for DB2 product
Features and benefits of RECOVER PLUS

- BMCSORT technology

- DB2 Solution Common Code (SCC) technology
  (a set of common components that several BMC DB2 products use)

Customers who acquire this solution benefit from all features of these products and technologies, as well as additional features that are available when one Recovery Management component can rely on the presence of all others. For more information, see the Recovery Management for DB2 User Guide.

RECOVER PLUS is also a component of the Administrative Assistant for DB2 solution and the Database Administration for DB2 solution.

Features and benefits of RECOVER PLUS

RECOVER PLUS offers significant features and benefits.

Reduced elapsed time required to recover

RECOVER PLUS offers several features that reduce the elapsed time that is required for recovery. For example, RECOVER PLUS can invoke a MERGE routine that concurrently uses a full image copy, incremental copies, a change accumulation file, and sorted log to efficiently re-create a current table space image. Then, the MERGE routine unloads index keys for a rebuild. The MERGE concept is exclusive to RECOVER PLUS and eliminates the need to read the table space page multiple times.

To further enhance the MERGE concept, RECOVER PLUS can run multiple log sorts and parallel MERGE phases in subtasks. You use the MAXLSORT option in the installation options or on the OPTIONS command to specify the maximum number of log sorts that can run concurrently. MAXLSORT also determines the number of MERGE/RESTORE/SNAP phases that can run in parallel, whether or not log records are processed.

RECOVER PLUS can perform multitasking index rebuilds in which multiple index key sorts and multiple index rebuilds are executed in parallel subtasks. You can define the level of concurrency to improve recovery performance by using the MAXKSORT option in the installation options or on the OPTIONS command. RECOVER PLUS can extract keys from partitions in parallel if the partitions are not being recovered. You can also use the KSORTSHARE option, available as an installation option or on the OPTIONS command, to improve index recovery. KSORTSHARE specifies if key sorts are shared among RECOVER PLUS table space recoveries (MERGE phases) running in parallel.
RECOVER PLUS can use the RECOVER UNLOADKEYS and RECOVER BUILDINDEX commands to rebuild a nonpartitioned index in a two-step process. RECOVER UNLOADKEYS can run in several jobs to extract and sort the keys greatly reducing overall elapsed time, work data set requirements, or both. RECOVER BUILDINDEX uses the keys from RECOVER UNLOADKEYS to build the index.

RECOVER PLUS considers all requests in the SYSIN data set and uses the most efficient technique for recovering all specified objects. This optimization process allows RECOVER PLUS to achieve efficiencies such as extracting keys while a table space is written, detecting stacked tape inputs, and automatically using the inputs in order.

RECOVER PLUS can use copies and log records to recover indexes, avoiding costly key sorts. Indexes can also be recovered with log only after being restored outside of DB2.

By using the BACKOUT option, RECOVER PLUS can recover to a point in time without image copies by using the table spaces, index spaces, and log records to return to a prior state. This process reduces resource consumption in several ways:

- Only the log between the point in time specified and the current log point is read and processed, which may substantially reduce log processing.

- In addition, the spaces are read for log backout processing, and image copy processing is completely eliminated.

- If indexes are not rebuilt and output image copies are not requested, only the pages that are updated by log records are read and written, which can greatly reduce I/Os and speed up the recovery.

RECOVER PLUS can recover by using Instant Snapshot copies, which are data set level copies made by the BMC COPY PLUS product with the BMC EXTENDED BUFFER MANAGER (XBM) product. Instant Snapshots are non-standard, point-in-time copies that are made on intelligent storage devices.

When you use RECOVER PLUS as part of the Recovery Management for DB2 solution, RECOVER PLUS can recover DB2 table spaces and indexes to any point in time and resolve inflight transactions at that point.
Better audit and recovery process control

RECOVER PLUS provides early recall of migrated data sets and the ability to make up to four copies of the recovered space and to register one or more of them.

When you use RECOVER PLUS for a point-in-time recovery, RECOVER PLUS allows generic keywords, such as TOCOPY LASTCOPY or TOLOGPOINT LASTQUIESCE, to identify the target copy or log point.

RECOVER PLUS can generate a list of archive, image copy, and change accumulation file tapes required for recovery. A report that lists all volumes of tapes and cartridges that are dynamically allocated during recovery is also available.

RECOVER PLUS provides a command to scan logs and provide estimates to help plan your recovery.

The RECOVER PLUS SIMULATE option, which is available if you have a valid Recovery Management for DB2 solution password, allows you to exercise all recovery resources. This option performs all non-destructive operations of the recovery to validate that all image copies, logs, and change accumulation files are available and usable.

Expanded recovery capabilities

RECOVER PLUS provides expanded recovery capabilities that enable you to recover from a dropped table space, create image copies without reading the table space or index space, and migrate data to other DB2 systems. With RECOVER PLUS, you can perform object ID translation, use unregistered input copies, and run recovery without SYSLGRNX entries.

Object ID translation

You can use the RECOVER PLUS OBIDXLAT option to translate internal DB2 object IDs (OBIDs) while processing. This feature supports RECOVER PLUS’s drop recovery function to allow recovery of dropped table spaces and allows translation during data migration. Improved migration checking when the INCOPY option is used detects incompatible migrations. Some examples of incompatible migrations are:

- migration of a partitioned table space into a nonpartitioned table space
- migration into a table space with a different page size
- migration of a segmented table space into a table space with a different segment size
RECOVER PLUS translates OBIDs and resets log point values in the pages, but in all other respects the internal structure of the source, target table spaces, and indexes must be the same.

When you use Instant Snapshots for migration and specify the OBIDXLAT option, RECOVER PLUS automatically detects if the OBIDs have not changed and avoids the overhead of reading and writing all of the pages.

For more information about the drop recovery function, see Chapter 7, “Recovering a dropped table space or table.”

**Optional use of non-registered input copies**

The INCOPY option allows you to specify the following types of copies as input:

- copies that have been dropped from SYSIBM.SYSCOPY
- copies made with DSN1COPY
- Instant Snapshot copies that are not registered in BMXCOPY
- copies that were made on another subsystem (or data sharing group)

Using non-registered input copies supports the following important functions of RECOVER PLUS:

- migration of data
- recovery of dropped table spaces

**Optional use of non-registered table space reset events**

Certain events that are recorded in SYSCOPY allow recovery of table spaces to be performed solely from the log. These events include LOAD LOG YES with REPLACE or REORG LOG YES. If these events no longer exist in SYSCOPY because of a table space being dropped or modified, the INLOG RBA/INLOG LOGPOINT option can be used to specify the starting point of a log-only recovery. The table space will be reset, and the log will be used to rebuild the table space. This feature does not apply to indexes.

**Running without SYSLGRNX records**

The NOSYSLGRNG option allows RECOVER PLUS to scan the log without referencing the DB2 SYSLGRNX table when processing a table space. The log is scanned from the beginning log point, such as the log point of the last copy, to TORBA/TOLOGPOINT or current. For indexes with the COPY NO attribute, index log records are always read without referencing the SYSLGRNX table.

This feature supports the RECOVER PLUS drop recovery function. This feature is also useful if you have a damaged SYSLGRNX table.
OUTCOPY ONLY

You can use the OUTCOPY ONLY option to build image copies without reading the table or index space or interfering with DB2 update access in any way. This feature allows you to avoid DASD contention and supports data migration.

Recovery test capability

The RECOVER PLUS simulation feature allows you to test DB2 recovery without performing any destructive operations or affecting the availability of the target table space or index. The Recovery Management for DB2 solution employs this feature to simulate a Disaster Recovery test.

TRANSFORM and High-speed Structure Change

Use the RECOVER PLUS TRANSFORM syntax to specify that you want to do a DDL transformation.

NOTE

TRANSFORM is part of the High-speed Structure Change process that includes the following methods:

- the SHRLEVEL REFERENCE method that requires either a Recovery Management password, or COPY PLUS and EXTENDED BUFFER MANAGER (XBM) passwords or COPY PLUS and SNAPSHOT UPGRADE FEATURE (SUF) passwords for the creation of the initial image copies as well as a RECOVER PLUS password for the TRANSFORM operation
- the SHRLEVEL CHANGE method that requires a Recovery Management password because of the use of Online Consistent Copies

The High-speed Structure Change process allows you to perform the following transformations:

- simple table space to partitioned by growth (PBG)
- non-LARGE partitioned table space to LARGE partitioned table space
- partitioned by range (PBR) table space to universal table space (UTS) PBR
- partitioned by range table space to UTS PBG

The High-speed Structure Change process also allows you to transform your indexes, avoiding costly sort CPU that would be required to rebuild the index.
At a more granular level, you can use the High-speed Structure Change process to simply change the segment size of a table space. This includes transforming a non-segmented table space into a segmented one. You can also change the data set size of your table space or index. This is helpful if you are reaching a capacity limit, i.e., exceeding the data set size or the number of data sets.

RECOVER PLUS performs the page transformations and uses the TRANSFORM option (page 171), the DSIZE option (page 205), and the PIECESIZE option (page 205).

See the *Recovery Management for DB2 User Guide* for more information.

## RECOVER PLUS operation

To accomplish recovery tasks, RECOVER PLUS offers the following features:

- uses the same DB2 resources that are used by the DB2 RECOVER utility
- uses any available change accumulation files as input to forward recovery
- uses index image copies that are made by COPY PLUS for DB2, the IBM DB2 COPY utility, RECOVER PLUS, or DSN1COPY
- uses Instant Snapshot copies that are made by COPY PLUS
- uses encrypted copies that are made by COPY PLUS
- uses cabinet copies that are created by COPY PLUS as part of the Recovery Management solution
- uses Online Consistent Copies, which you can create if you have the Recovery Management solution
- supports a wide range of recovery strategies by using a variety of commands and options

During a recovery, RECOVER PLUS automatically optimizes recovery processing by performing the following tasks:

- optimizing block sizes
- extracting index keys, when possible and requested, during table space recovery
detecting and ordering activities for the proper positioning of stacked tape data sets, change accumulation files, and input and output copies containing all partitions of a partitioned space

- using a subtask to preallocate and open VSAM data sets required during recovery

- automatically falling back, without restarting, to a previous copy or Instant Snapshot, if an invalid or unavailable image copy or Instant Snapshot is encountered during recovery

Log data sets and table space data sets are dynamically allocated to the job. RECOVER PLUS also dynamically allocates the input image copy data sets and required change accumulation files. The target table and index spaces are identified through IBM-compatible syntax, and the target DB2 subsystem is identified in the EXEC statement utility parameters. As an option, RECOVER PLUS provides support for dynamic allocation of output image copies.

RECOVER PLUS parses all RECOVER PLUS command statements before initiating recovery. RECOVER PLUS constructs a recovery plan and produces a report on this plan. Other options allow control over other aspects of the recovery. For example, you can specify the device type (SORTDEVT) and number of temporary data sets (SORTNUM) for sorting, or you can specify the MAXKSORT option for multitasking index key sorts and index rebuilds. You can further refine your index recovery using the KSORTSHARE option. You use the MAXLSORT option to specify concurrent log sorts and parallel MERGE phases. RECOVER PLUS provides the ANALYZE ONLY option to view recovery resources and preview recovery plans without actually executing a recovery job. RECOVER PLUS also provides a LOGSCAN command to gather information about the recovery and provide information that you can use to choose sort parameters.

RECOVER PLUS uses the BMC BMCSORT technology for sorts. BMCSORT provides RECOVER PLUS with more control of the sort process than external sort routines provide. This added control helps prevent memory-related problems during the sort process. RECOVER PLUS allocates the amount of resources to each sort process based on the amount of work that RECOVER PLUS determines that the sort process will perform. RECOVER PLUS also dynamically detects excess available memory and allocates a percentage of it to the sort processes.

The recovery simulation feature, which requires a Recovery Management for DB2 solution password, provides a non-destructive mechanism to exercise all recovery resources, including image copies, log records, and change accumulation files. This feature allows customers to ensure that all recovery resources are available to recover a table space or index without actually recovering it.
RECOVER PLUS controls spaces by altering the DB2 status of the objects as necessary. Synchronization with BMC utility operations is provided through DB2 tables that are delivered and installed as part of RECOVER PLUS.

RECOVER PLUS does not use the DB2 subsystem buffers; however, it uses the DB2 subsystem to find information for the recovery process.

Using IBM DB2 resources

RECOVER PLUS is fully compatible with IBM utilities and can rely on the same recovery resources and information during a recovery. The DB2 resources that are used during a typical RECOVER PLUS execution include the following items:

- full image copy
- incremental copies
- active and archive logs
- log ranges that are recorded in the SYSIBM.SYSLGRNX table
- information that is stored in the DB2 catalog

**NOTE**

RECOVER PLUS does not support DB2 concurrent copies.

RECOVER PLUS, COPY PLUS and REORG PLUS use an additional repository of information about recovery resources that are not used by the DB2 RECOVER utility. These additional resources include the following items:

- image copies of a COPY NO index
- data set level image copies of nonpartitioned COPY YES indexes
- incremental index copies
- Instant Snapshot image copies
- encrypted copies
- Online Consistent Copies (Recovery Management solution)
- cabinet copies (Recovery Management solution)
Using RECOVER PLUS with RECOVERY MANAGER

You can use the BMC RECOVERY MANAGER for DB2 product with RECOVER PLUS. RECOVERY MANAGER provides an automated way to generate complex JCL for optimal recovery jobs and includes a facility to optimize multiple recovery jobs.

RECOVERY MANAGER provides multiple ways to designate groups of objects, including defined change accumulation groups and OBJECTSETs.

RECOVERY MANAGER creates groups using OBJECTSET. However, you cannot use RECOVERY MANAGER to build the RECOVER PLUS JCL to use OBJECTSET. For RECOVER PLUS, you must add OBJECTSET to the SYSIN manually. For information about the OBJECTSET syntax, see the OBJECTSET descriptions in Chapter 3, “RECOVER PLUS syntax.”

For complete details about the RECOVERY MANAGER product, see the RECOVERY MANAGER for DB2 User Guide.

Using RECOVER PLUS with R+/CHANGE ACCUM

The BMC R+/CHANGE ACCUM for DB2 product, with its ACCUM command, offers the ability to create new recovery resources called change accumulation files. When used during a RECOVER PLUS execution, change accumulation files significantly streamline normal and disaster recovery processes. R+/CHANGE ACCUM enhances the speed at which recovery processes take place by extracting in advance the log record data from the DB2 log that is required for recovery. Indexes supported with COPY PLUS copies and COPY YES indexes may also have log changes accumulated.

When R+/CHANGE ACCUM is available, RECOVER PLUS can process ACCUM and RECOVER command statements in the same step for maximum efficiency in log processing.

R+/CHANGE ACCUM is automatically installed with RECOVER PLUS but requires its own password for use or a Recovery Management solution password. (R+/CHANGE ACCUM is part of the Recovery Management solution.)

For more information about R+/CHANGE ACCUM, see the R+/CHANGE ACCUM for DB2 User Guide.
RECOVER PLUS limitations

RECOVER PLUS does not support the following items:

- DB2 Version 7 or earlier
- recovery of catalog or directory spaces
- recovery by using the ERROR RANGE and PAGE options (however, RECOVER PLUS removes an existing error range after a successful table space or index recovery)
- setting the check pending flag on a space recovered to a point in time prior to the definition of a column constraint
- recovery of the DB2 objects that are used by RECOVER PLUS:
  - xxxxxx.CMN_BMCUTIL
  - xxxxxx.CMN_BMCSYNC
  - xxxxxx.CMN_BMCXCOPY
  - xxxxxx.CMN_BMCLGRNX
  xxxxxx is the creator name or ID.
- index logical partition recovery
- setting check pending flags for related spaces that are not recovered
- following uses of an inline copy:
  - use of an inline copy as a source when the OUTCOPY ONLY option is used
  - use of an inline copy that has been merged with subsequent incremental copies

The product tries to detect this condition by examining rows in SYSIBM.SYSCOPY indicating that an inline image copy is registered at the same START_RBA as a previous incremental copy. However, if those entries have been removed by the DB2 MODIFY utility or the COPY PLUS MODIFY command, other errors may occur.

- use of an Instant Snapshot copy with the OUTCOPY ONLY option
RECOVER PLUS limitations

- **REBUILD INDEX** command if all of the following conditions are true:
  
  — A table is altered to add a column with the ROWID attribute.
  — An index is created with this new column.
  — A REORG is not performed.

- use of the second copy of archive logs, as directed by the DSNZPARM ARC2FRST parameter

---

**NOTE**

The same facility is provided with the RESOURCE SELECTION LOGS option of RECOVER PLUS.

---

- tracker sites
- LISTDEF processing
- inline statistics for REBUILD
- same order of use for volumes from multivolume STOGROUP-defined spaces as DB2 RECOVER

RECOVER PLUS might use volumes from multivolume STOGROUP-defined spaces in a different order than DB2 RECOVER. You can control the order with the STOGROUP...USEORDER option.

Consider the following restrictions when working with RECOVER PLUS and DB2 Version 9:

- **RECOVER PLUS** calls DSNUTILB for the following DB2 Version 9 features:
  
  — indexes defined using an expression

  Some rebuild requests for indexes defined using an expression are passed to DSNUTILB. Some are processed natively, including, SUBSTR, UPPER, and LOWER. Because some rebuild requests are not supported natively, RECOVER UNLOADKEYS does not support indexes defined using an expression.

  However, recover requests are handled by RECOVER PLUS.

  — indexes with RANDOM-order columns

  Rebuild requests for indexes with RANDOM-order columns are passed to DSNUTILB, but recover requests are handled by RECOVER PLUS.
When you are performing a drop recovery or a migration, the source definition must match the target definition, including the number of partitions. For partition-by-growth universal spaces, if the number of partitions for the source has grown to more than one partition, the source would no longer match the target, and the drop recovery or migration fails.

- RECOVER PLUS versions earlier than 8.1.00 will not work with DB2 Version 9.

- RECOVER PLUS versions earlier than 10.1.00 will not work with DB2 Version 10.

- If you recover an object to a point in time using DB2 Version 9 (or later) DSNUTILB, you need to use DSNUTILB for subsequent recoveries of this object. You can recover this object using RECOVER PLUS after you perform one of the following actions:

  - make an image copy
  - use REORG LOG YES
  - use LOAD REPLACE LOG YES

### Planning recovery

RECOVER PLUS is a powerful tool that aids in the speedy recovery of DB2 objects, but you are still responsible for recovery planning. You should become familiar with features and options that are offered by the utility and plan recovery strategies for different situations.

The features of RECOVER PLUS have been designed with many different types of recovery situations in mind. Review all of the pertinent documentation and plan for media failures, application backouts, software failures, and other recovery scenarios that may occur.

When you are planning for recovery, consider using RECOVERY MANAGER for DB2 to generate JCL in advance, manage emergencies, and plan for disaster recovery.

**NOTE**

The Recovery Management for DB2 solution integrates the features of RECOVER PLUS and RECOVERY MANAGER as well as other backup and recovery products for DB2 described in “Solution integration” on page 45. With this solution, you can benefit from all features of these individual products and additional features that are available when one Recovery Management component can rely on the presence of all others.
Recovery strategies

With RECOVER PLUS you can control recovery with options that cover a wide range of recovery scenarios. These recovery scenarios are implemented by combining one or more RECOVER PLUS commands and their options.

**NOTE**

All references to the RECOVER INDEX command or index recovery refer to the use of image copies and log. The REBUILD INDEX command or index rebuild refers to extracting and sorting index keys and building indexes. For more information, see the description of the INDEXLOG option (page 541).

Using LOGSCAN for estimates

To help in planning for recovery, RECOVER PLUS provides the LOGSCAN command. LOGSCAN gathers vital information about the log records that you can use to determine a recovery strategy. LOGSCAN uses only the LOG INPUT phase and writes the output to AFRPRINT or to a specified ddname (SYSSCAN). RECOVER PLUS performs no log sort when you use LOGSCAN.

**NOTE**

The LOGSCAN command does not consider change accumulation files.

Output from the LOGSCAN command consists of

- type of the object
- name of the object
- partition number (0, if nonpartitioned)
- number of log records
- byte count
- calculated NUMREC EST value and AVGRECSZ

**NOTE**

The LOGSCAN output does not include statistics for index rebuilds.

For more information, see “LOGSCAN” on page 244 and “Example 13: Using the LOGSCAN command” on page 387.
Using BACKOUT for point-in-time recovery without image copies

When you use RECOVER PLUS to do a point-in-time recovery with the TOLOGPOINT option and the spaces are undamaged and current, you can use the BACKOUT option to employ a technique that uses the spaces and log records to restore spaces to a prior state. Only the log between the point in time specified by TOLOGPOINT and the current log point is read and processed with this technique, which may substantially reduce log processing. In addition, the spaces are read for log backout processing, and image copy processing is completely eliminated. If indexes are not rebuilt and output image copies are not requested, only the pages updated by log records are read and written, which can greatly reduce I/Os and speed up the recovery. (For more information, see “Using copies and logs for index recovery” on page 62.)

To convert an existing partial recovery job to this technique, simply add the BACKOUT YES option to the usual syntax. Add the INDEXLOG YES or INDEXLOG AUTO option (note that INDEXLOG AUTO requires use of a Recovery Management password) and RECOVER PLUS assumes that the related indexes should be backed out to the same point in time. For example:

```
OPTIONS BACKOUT YES INDEXLOG YES
RECOVER TABLESPACE PAYROLL.EMPLOYEE
Recover TABLESPACE PAYROLL.RULES
TOLOGPOINT LASTQUIESCE
RECOVER INDEX (ALL) TABLESPACE PAYROLL.EMPLOYEE
RECOVER INDEX (ALL) TABLESPACE PAYROLL.RULES
```

When you select this strategy for potential point-in-time recovery, image copies may be avoided prior to batch runs without risking lengthy recovery times. Image copies do not need to be made of indexes to use the BACKOUT technique. (Indexes would still need to be rebuilt for media failure or disaster recovery.)

**NOTE**

A recovery that uses the BACKOUT option uses only the regular DB2 logs. A BACKOUT recovery does not use logs from the change accumulation files.

After a backout recovery, you must make a copy of the index before a recovery to current (by using a copy and log records) can be done. To maintain the ability to recover from log in case of a future media error, you can start a COPY PLUS SHRLEVEL CHANGE copy running in parallel with other applications, as soon as the backout recovery is complete.
Comparing BACKOUT performance

You can use the OUTCOPY YES option in your BACKOUT run. However, this request may degrade performance, because making an output copy causes all pages of the index space to be read.

**NOTE**

If you forego making a copy, you assume the risk that you will have to rebuild the index if it becomes damaged before your next image copy.

For more information, see “Using BACKOUT” on page 426.

**Comparing BACKOUT performance**

The backout recovery strategy is dramatically faster than most traditional forward recoveries as shown in Figure 1 on page 61. A backout recovery does not require image copies to perform a point-in-time recovery. Instead, it backs out the log records to undo or redo the changes that occurred between the selected point in time and the current point. This method returns the spaces and indexes to the required state without the overhead of restoring image copies, or rebuilding or restoring indexes.
Figure 1  Point-in-time recovery (forward recovery versus BACKOUT)

Using BACKOUT AUTO recovery

When you request a point-in-time recovery, using the BACKOUT strategy is desirable. However, there are some restrictions that prevent an object from being backed out. The BACKOUT to forward strategy performs a BACKOUT on all eligible objects and automatically schedules a forward recovery for the ineligible objects, as well as any other objects that failed in the BACKOUT recovery step.

In RECOVER PLUS version 9.2.00 and later, the BACKOUT installation option (page 556) defaults to AUTO. With this installation option set to its default value, if you are using a Recovery Management password, point-in-time recoveries automatically use the BACKOUT AUTO recovery strategy. If you are not using a Recovery Management password, point-in-time recoveries use a standard forward recovery. You can also request the BACKOUT to forward recovery strategy using the BACKOUT AUTO option (page 125) on the OPTIONS command in the JCL.
Using copies and logs for index recovery

RECOVER PLUS supports index recovery from full image copies, incremental image copies of indexes made by COPY PLUS, logs, and change accumulation files. For traditional forward recoveries, you can make the required full image copies with COPY PLUS for DB2 or the IBM COPY utility. You can make incremental image copies of indexes with COPY PLUS. RECOVER PLUS can also use a copy that is produced with DSN1COPY when you specify the INCOPY...LOGPOINT keywords.

Point-in-time index recovery that uses logs as described above is also supported, allowing applications to be reset to a prior point in time without sorts of index keys. To use this feature, you must use the INDEXLOG YES or INDEXLOG AUTO option (as an installation option or OPTIONS command option) or the RECOVER INDEXSPACE command.

Performing forward recovery without an image copy

RECOVER PLUS allows forward recovery from the log without using a copy to restore the space. You accomplish this type of recovery by using the RECOVER TABLESPACE, RECOVER INDEXSPACE, or RECOVER INDEX command (with INDEXLOG set to YES or AUTO in the installation options or on the OPTIONS command) and the LOGAPPLY ONLY option or the LOGONLY option. This feature allows you to restore a table space or index from a non-DB2 backup. The LOGAPPLY ONLY option also allows you to make forward adjustments to a table space or index after a point-in-time recovery.

Recovering with Instant Snapshot copies

RECOVER PLUS supports forward recovery by using Instant Snapshot copies that are made by the BMC COPY PLUS product with the BMC EXTENDED BUFFER MANAGER (XBM) product. Instant Snapshot copies are data set level copies of table spaces or indexes that are made by using intelligent storage devices. These nontraditional copies are registered in the BMCXCOPY table, not in
SYSIBM.SYSCOPY. Only full copies are allowed. To recover these copies, specify XBMID in the installation options or on the OPTIONS command. Additionally, the SNAPSHOT option adds support for Instant Snapshots with the INCOPY FULL syntax of the RECOVER command.

For more information, see “Using Instant Snapshots” on page 429.

Using parallel MERGE phases and log sorts

RECOVER PLUS allows multiple log sorts and parallel MERGE phases to run concurrently in subtasks, which can provide an improvement in recovery performance. For each table space, the log sorts are distributed over the number of sorts specified by the MAXLSORT option (page 138 and page 553). MAXLSORT also determines the number of MERGE phases that can run in parallel. When you combine index key sorts (see “Using multitasking index key sorts and index rebuilds”) with log sorts, the KSORTSHARE option (page 139 and page 554) determines whether RECOVER PLUS shares the key sorts among table space recoveries (MERGE phases) running in parallel.

For more information, see “Setting the MAXLSORT and KSORTSHARE options” on page 441.

Using multitasking index key sorts and index rebuilds

RECOVER PLUS allows multiple key sorts and multiple index rebuilds to run concurrently in subtasks, which can provide a significant recovery performance improvement. For each table space, index keys for all indexes being rebuilt are distributed over the number of sorts specified by the MAXKSORT option (page 136 and page 544). When all keys are input to the sort for a group of indexes, the build for each index is performed in a subtask. One subtask is used for each group of indexes that were previously sorted.

The KSORTSHARE option, mentioned in “Using parallel MERGE phases and log sorts,” can also improve your index recovery.

For more information, see “Setting the MAXKSORT option” on page 439.
Eliminating interim data sets for index keys

By using the NOWORKDDN option when you are rebuilding indexes, you can eliminate writing index keys to an intermediate data set before sorting. Use of the data set can also be eliminated if you do not code the WORKDDN option and you do not include a SYSUT1 DD statement in your JCL. Eliminating the interim data set allows RECOVER PLUS to have BMCSORT run as a background task while the keys are extracted. Although this choice limits restart options, it improves the normal recovery time when rebuilding indexes.

Rebuilding nonpartitioned indexes on a partitioned table space

Another powerful execution option of the RECOVER PLUS utility is the RECOVER UNLOADKEYS command, which constructs data sets that are used by the RECOVER BUILDINDEX command to rebuild nonpartitioned indexes on a partitioned table space. The RECOVER UNLOADKEYS command permits extraction of index keys from partitions in separate, concurrent jobs. You can set up multiple jobs to do the processing on the individual parts followed by a job to rebuild the nonpartitioned index. This approach runs in less time than a single job that reads all of the partitions and rebuilds the index.

Each unloaded keys data set is sorted individually in RECOVER UNLOADKEYS. These sorted files are merged by RECOVER BUILDINDEX, which results in multiple smaller sorts that could improve performance and reliability in some cases. This method may be particularly useful when you are working with very large table spaces.

Using resource selection

RECOVER PLUS resource selection features provide a way for you to control how RECOVER PLUS accesses and orders the multiple images of the DB2 log, image copies, and BMC change accumulation files during recovery. These powerful features provide additional control, which you can use to avoid resources that are unavailable, avoid contention, and use copies registered for one site type while running as another site type.

For example, to avoid a single point of failure, DB2 subsystems are generally run with dual active logs. Dual archiving is available as well. In most cases, RECOVER PLUS attempts to use Active 1 and then Active 2, followed by Archive 1 and Archive 2. The first available file that contains the log records required is actually read. You can use the RECOVER PLUS resource selection features to control whether, and in what order, to use each source of log records.
You can create multiple copies of image copies and change accumulation files. You can create two pairs, one designated LOCAL and one REMOTE. These copies are classified as the following types:

- LOCAL PRIMARY
- LOCAL BACKUP
- REMOTE PRIMARY
- REMOTE BACKUP

Normally, RECOVER PLUS uses only LOCAL copies if the DB2 subsystem is running as a LOCAL site and only REMOTE copies for the REMOTE site. PRIMARY is normally considered before BACKUP unless the TOCOPY option indicates a backup. You can use the RECOVER PLUS RESOURCE SELECTION option to control which copies to attempt to use, and in what order, regardless of whether they are registered as LOCAL or REMOTE.

**Previewing recovery activity**

You can use the RECOVER PLUS ANALYZE ONLY option to preview recovery activity before executing a recovery. ANALYZE ONLY reports information about image copy data sets, log data sets, change accumulation files, and log ranges that RECOVER PLUS will use for recovery. RECOVER PLUS includes an optional tape pick list and recall list in its reporting.

Information about the record sizes for work files and index sorts is also provided. You can use these statistics to accurately allocate space for log sort and index sort work files and index key work data sets. As an option, you can recall archived resources during an ANALYZE ONLY run.

This feature provides some predictability of recovery operations by providing to you the means to examine the recovery procedures and strategies that you wish to implement.

The LOGSCAN command provides information about the number and size of log records required for recovery. You can use this information to size the log sort. A printed report describes the log records for each object that is named in the LOGSCAN command.
Testing recovery with simulation

You can use the SIMULATE YES option on the OPTIONS command (page 134) to go a step beyond previewing recovery activity to testing it. This non-destructive option performs most of the work of a normal recovery. After reading and merging image copies, log files, and change accumulation files, RECOVER PLUS discards the output and leaves the underlying table space unaffected. Using this option, you can see exactly how a recovery will run without sacrificing data availability.

NOTE
Simulation is a feature of the Recovery Management for DB2 solution and requires a valid Recovery Management solution password.

Recovery simulation provides a way for you to validate that you can recover your application data. You can use recovery simulation to verify that needed recovery resources are valid and available and that log apply can be performed.

The following synonyms are required in RECOVER PLUS syntax when you specify OPTIONS SIMULATE YES:

- SIMRCVR TABLESPACE, INDEXSPACE, or INDEX for RECOVER TABLESPACE, INDEXSPACE, or INDEX (page 167)
- SIMRBLD INDEX for REBUILD INDEX (page 214)
- SIMRCVR UNLOADKEYS for RECOVER UNLOADKEYS (page 228)
- SIMRCVR BUILDINDEX for RECOVER BUILDINDEX (page 237)

You can use RECOVERY MANAGER for DB2 to build the JCL for recovery simulation. For more information, see the RECOVERY MANAGER for DB2 User Guide.

Operational considerations

When you specify OPTIONS SIMULATE YES in a RECOVER PLUS job, RECOVER PLUS proceeds with all recovery activities that are not destructive. Destructive actions that could affect real database objects or the recovery environment are simulated or suppressed. The following activities are included:

- reading and sorting log records
- merging image copies, log records, and change accumulation files
- extracting index keys
RECOVER PLUS does not stop any table spaces or indexes during simulation mode.

RECOVER PLUS performs the following tasks for table spaces and for indexes:

- reads full and incremental image copies that are required for recovery
- reads change accumulation files that are required for recovery
- simulates applying log records, change accumulation log records, or both
- extracts index keys for any index rebuilds that are required

RECOVER PLUS processes the extracted, sorted keys and builds each index page in storage.

RECOVER PLUS does not perform the following activities:

- write any output pages to DASD
- write any output image copies
- write to BMCSYNC or BMCUTIL
- register any events in SYSIBM.SYSCOPY
- set or reset any pending flags

**Limitations**

Use of recovery simulation has the following limitations, most of which result from not performing any disruptive activities on the spaces that are involved.

- The following RECOVER PLUS features are not supported in recovery simulation:
  - recovery from Instant Snapshot copies
  - BACKOUT recovery
  - LOGONLY or LOGAPPLY ONLY
  - ANALYZE ONLY or ANALYZE SCANONLY
  - INLINE image copies
  - LOGSCAN
  - point-in-time recovery

**NOTE**

Simulation of point-in-time recovery is indirectly supported by specifying a log point with the SIMULATE option. In this case, RECOVER PLUS simulates recovery of all objects in the SYSIN to the same log point. Simulation of point-in-time recovery is also provided by the RECOVERYPOINT option if the Recovery Management for DB2 solution is installed.

- Simulation runs are not restartable.
The requested recovery scenario can require unloading keys from the real table space (partition) if REBUILD or UNLOADKEYS is requested for the table space (partition) not recovered in the same step. Such unload operations involving real table spaces (partitions) will be suppressed in simulation mode.

Objects may be in any initial status and there is no impact on the status or contents of the real database objects.

Recovering while activity continues

Activity can continue on the original table space while a recovery is being performed to a consistent or arbitrary point on a copy of the table space. You perform this task by naming new data sets for the recovery of the DB2 table space that is subject to errors.

After recovery is complete, apply the log to make the new table space current. Renaming the new data sets will cause transactions to access the new table space.

BMCXCOPY registrations

Full image copies of indexes are registered in the BMC–supplied BMCXCOPY table if one of the following situations applies:

- The index is defined with COPY NO.
- The index is a nonpartitioning index with data set level (DSNUM n) copies, even if the nonpartitioning index is defined as COPY YES.

These registrations are made in BMCXCOPY rather than in SYSCOPY because DB2 does not allow registration of such image copies in SYSCOPY.

Incremental index copies are always registered in BMCXCOPY.

Instant Snapshot copies of table spaces and indexes, which are made by the BMC COPY PLUS utility with the BMC EXTENDED BUFFER MANAGER (XBM) product, are registered in the BMCXCOPY table with an STYPE of V. When a local primary Instant Snapshot copy is registered in BMCXCOPY and a local backup copy is made, the local backup copy is also registered in BMCXCOPY, even if it is a regular DB2 image copy.

Encrypted image copies, which are made by the BMC COPY PLUS utility, are registered in the BMCXCOPY table with an STYPE of e. For more information about encrypted copies and their registration, see “Recovering encrypted copies” on page 462.
When ICTYPE=P, BACKOUT recovery is registered in the BMCXCOPY table with an STYPE of B.

Cabinet copies, which are made by the Recovery Management solution, are registered in BMCXCOPY with a COPY_TYPE value of C. For more information about cabinet copies, see “Using cabinet copies” on page 73.

The BMCXCOPY table is described in Appendix C, “BMC utilities database.”

Creating copies from recovery resources

You can use RECOVER PLUS to create and register table space or index image copies based on other recovery resources without accessing or affecting the availability of the object. For example, you can create image copies from other copies, logs, and change accumulation files.

This type of copy can be created as of a prior point in time or at the current time. By creating copies from other recovery resources, you can create copies registered at the point in time that an application started while the application is running. If you must recover to that point, you will have a registered copy available. If you need a copy at a given point in time, you can create and register the copy as needed.

Overriding registered copies

If you have valid copies that are not registered in SYSIBM.SYSCOPY or the BMCXCOPY table, you can use RECOVER PLUS to override the copies that are specified in these tables with a list of one or more copies to be used instead.

Recovering LOB spaces with RECOVER PLUS

RECOVER PLUS recovers or rebuilds all components of LOB table spaces and indexes. The BACKOUT function is not available for LOB auxiliary table spaces.
Recovering LOB spaces with RECOVER PLUS

Recovery of LOG NO LOB auxiliary table spaces

When RECOVER PLUS recovers a LOB defined as LOG NO, RECOVER PLUS leaves the space in AUXW status if it detects that there were any unlogged updates in the range of the recovery. Any LOB objects that were updated with unlogged updates are invalidated and reported by the message BMC96124W:

BMC96124W LOG NO UPDATE FOR ROWID(X'50633BFCD66BD08B210401581C30404040'), VERSION 2

DB2 allows SQL access to all other rows in the table even though the space is in AUXW status. DB2 also allows SQL access to all other columns of the affected row other than the invalidated LOB. Also, DB2 allows SQL to update the LOB value itself to a new value.

The format of the BMC96124 message allows for an easy cut and paste to SQL using the ROWID value as a predicate as shown in the following statement:

```
SELECT EMPNO,FNAME,LNAME FROM WRPDB83.WRPTB83A
WHERE ROWID = ROWID(X'50633BFCD66BD08B210401581C30404040')
```

```
+---------+---------+---------+
| EMPNO   | FNAME    | LNAME    |
|---------+---------+---------|
| 1       | JOHN     | SMITH    |
+---------+---------+---------+
```

DB2 allows SQL access to all other rows in the table even though the space is in AUXW status. DB2 also allows SQL access to all other columns of the affected row other than the invalidated LOB. Also, DB2 allows SQL to update the LOB value itself to a new value.

The format of the BMC96124 message allows for an easy cut and paste to SQL using the ROWID value as a predicate as shown in the following statement:

```
SELECT EMPNO,FNAME,LNAME FROM WRPDB83.WRPTB83A
WHERE ROWID = ROWID(X'50633BFCD66BD08B210401581C30404040')
```

```
+---------+---------+---------+
| EMPNO   | FNAME    | LNAME    |
|---------+---------+---------|
| 1       | JOHN     | SMITH    |
+---------+---------+---------+
```

Point-in-time recoveries

You should always recover a LOB base table space and its related auxiliary table space or spaces and indexes together and to a consistent point in time. Recovery of any individual component to any other point in time might leave the space in an error condition that is difficult to resolve.

If the space is left in any of the exception statuses described in the following list, use the DB2 CHECK DATA utility to remove the error status. Consult the DB2 documentation for instructions for using this utility.

- If RECOVER PLUS recovers the base table space to a previous point in time and does not recover the auxiliary table space to the same point in time, RECOVER PLUS leaves the base table space in ACHKP status.
- If RECOVER PLUS recovers the auxiliary table space to a previous point in time and does not recover the base table space to the same point in time, RECOVER PLUS leaves the base table space in ACHKP status and the auxiliary table space in CHKP status.
If RECOVER PLUS recovers the base and auxiliary table spaces to the same point in time, but that point in time is not a consistent point (such as a quiesce point), RECOVER PLUS leaves the base table space in ACHKP.

If RECOVER PLUS recovers the base and auxiliary table spaces to the same point in time, and if that point in time is a registered quiesce point, and if all related indexes are rebuilt or recovered to the same point in time, no error statuses are set.

If RECOVER PLUS recovers the base and auxiliary table spaces to the same point in time specified in OPTION RECOVERYPOINT syntax (available only with the BMC Recovery Management solution), and if all related indexes are rebuilt or recovered to the same point in time, no error statuses are set.

**Automatic fallback without restarting**

During the recovery of a space, fallback is the process of using an equivalent or earlier image copy to replace an invalid or unavailable image copy. Automatic fallback extends the RECOVER PLUS fallback capabilities of an initial execution to all situations involving standard image copies or Instant Snapshot copies, except for recoveries that use the INCOPY option or the RECOVER UNLOADKEYS command. The automatic fallback feature of RECOVER PLUS can save you time and increases the usability of the utility.

---

**NOTE**

Automatic fallback does not occur if you specify:

- INCOPY option on a RECOVER command
- RECOVER UNLOADKEYS command
- REBUILD INDEX command with the WORKDDN option

For more information, see “Fallback recovery with RECOVER PLUS” on page 424.

**Using incremental index image copies**

Using incremental copies of indexes made by COPY PLUS in recovery has the following benefits:

- reduces the need for excessive DB2 log application for an index space recovery
- reduces the amount of data that must be copied to disk or tape in order to maintain current copies
Using incremental index image copies

- reduces the volume of image copy data to manage
- improves recovery times compared to rebuilding indexes or to using only full index image copies

**NOTE**

To use the incremental index copies made with COPY PLUS version 8.1.00 and later, RECOVER PLUS version 8.1.00 or later is required. If you attempt a recovery using incremental index copies and earlier versions of RECOVER PLUS, unpredictable results can occur.

RECOVER PLUS recognizes incremental copies of indexes recorded in BMCXCOPY, includes them in RECOVER PLUS analysis and planning, shows them in the RECOVER PLUS object summary, and uses them whenever appropriate to speed up recovery of indexes.

When you use incremental index copies for recovery, note the following considerations:

- Incremental index space copies that are produced by COPY PLUS are registered in the BMCXCOPY table.

- Because index spaces lack the ability to identify modified pages, incremental index space copies are always produced using the READTYPE FULLSCAN method.

- Because index spaces lack the ability to identify modified pages, the RESETMOD option for copies does not apply. All index copies are RESETMOD NO.

- RECOVER PLUS processing for cumulative incremental image copies applies to incremental index copies. For more information, see “Recovery using cumulative incremental image copies” on page 90.

**Restrictions on incremental index copies**

The following restrictions apply to incremental index copies:

- You cannot use DSNUM \( n \) incremental copies of nonpartitioned indexes in a DSNUM ALL recovery.

- COPY PLUS and RECOVER PLUS do not support incremental index copies of catalog and directory.
Incremental copies of nonpartitioned indexes and recovery

When you make incremental copies for nonpartitioned indexes, you should make these copies using the COPY PLUS IXDSNUM=ALL option. If you make the copies using the COPY PLUS IXDSNUM=DATASET option, your RECOVER PLUS statements must specify the data set number on the RECOVER statement.

If you copy a nonpartitioned index with IXDSNUM=ALL, examples of your RECOVER PLUS syntax are:

\[
\text{RECOVER INDEX IX.A} \\
\text{or} \\
\text{RECOVER INDEX IX.A DSNUM ALL}
\]

If you copy a nonpartitioned index with IXDSNUM=DATASET, examples of your RECOVER PLUS syntax are:

\[
\text{RECOVER INDEX IX.A DSNUM } n \\
\text{or} \\
\text{RECOVER INDEX IX.A DSNUM } n:n \text{ (more than one data set to be recovered)}
\]

For more information about IXDSUM option, see the COPY PLUS for DB2 Reference Manual.

Using cabinet copies

RECOVER PLUS supports the cabinet copy feature of the Recovery Management for DB2 solution.

**NOTE**

Because the cabinet copy feature is part of the Recovery Management for DB2 solution, making cabinet copies and using them for recovery requires use of a valid Recovery Management solution password. For more information about cabinet copies and Recovery Management, see the Recovery Management for DB2 User Guide.

Cabinet copies provide a performance enhancement when you are copying a large number of spaces. In such cases, the overhead to open and close each copy data set can be a significant component of overall run time.
Cabinet copies allow you to copy all the spaces in a group into a single data set called a cabinet file. The cabinet file is allocated and deallocated only once, regardless of the number of objects that are copied to or recovered from the cabinet file. Because there is no file opening or closing for each space in the cabinet file, the file header and trailer records, including the EOF markers, are omitted from cabinet files, and performance is greatly improved. You can copy the cabinet files to either DASD or tape.

In addition to providing a performance enhancement, cabinet copies can save resources because using cabinet copies can:

- save disk space because of the efficient use of space within a cabinet file
- reduce the number of MVS catalog entries

RECOVER PLUS automatically detects and uses cabinet copies if they are available for recovery for any spaces in the cabinet copy. Phases to recover objects will be planned in the order that the object image copies appear in the cabinet copy unless this would conflict with other planning restrictions such as change accumulation files or output copies.

**Restrictions for cabinet copies**

The following restrictions apply to cabinet copies:

- A minimum z/OS level of 1.7 is required for cabinet copies made to disk.
- For a recovery using cabinet copies, you must use RECOVER PLUS version 8.1.00 or later.
- An image copy contained within a cabinet copy must have a row present in BMCXCOPY on the subsystem on which the recovery is run.
- The INCOPY option of RECOVER PLUS does not support cabinet copies.

**NOTE**

If you have only a cabinet copy and need to do a drop recovery, you can use the COPY PLUS COPY IMAGECOPY command to unstack a cabinet copy and register the copies in SYSCOPY. You can then use the individual copies with the INCOPY option.

- Online consistent copies, another feature of the Recovery Management solution, do not support cabinet copies.
- You cannot make cabinet copies using the Instant Snapshot functionality.
Registration of cabinet copies

Because cabinet copies can be recovered only by using RECOVER PLUS, they are registered in the BMCXCOPY table. A COPY_TYPE value of C indicates the copy is a cabinet copy. A row is created for each member within the cabinet copy and each member has the same DSNAME. The DSNAME is the name of the cataloged cabinet file as indicated in the COPY PLUS OUTPUT command. The cabinet members are individually registered in BMCXCOPY as uncataloged. The cabinet file is always cataloged.

NOTE

Copies made using DSNUM ALL reduce the number of entries in BMCXCOPY for spaces with many partitions.

Using Online Consistent Copies

Online consistent copies, copies made without any outage and that are physically and transactionally consistent, are a feature of the Recovery Management for DB2 solution and are registered in the BMCXCOPY table. These copies, which are used for recovery to copy, are also useful for migration. Version 8.1 of Recovery Management introduces the use of Online Consistent Copies and log apply for forward recovery.

NOTE

Because the Online Consistent Copy feature is part of the Recovery Management for DB2 solution, making Online Consistent Copies and using them for recovery requires use of a valid Recovery Management solution password. For more information about Online Consistent Copies and Recovery Management, see the Recovery Management for DB2 User Guide.

RECOVER PLUS supports the use of Online Consistent Copies and log apply for forward recovery to the current time or to a point in time, as well as recovery to a copy.

NOTE

Note the following considerations:

- RECOVER PLUS supports Online Consistent Copies with log apply when you do not specify INCOPY.
- RECOVER PLUS supports Online Consistent Copies with INCOPY TOCOPY specified.
- RECOVER PLUS does not support Online Consistent Copies when you specify both INCOPY and log apply (RBA or LOGPOINT specified in the INCOPY specification).
Solving other problems with RECOVER PLUS

Extra processing is required to make Online Consistent Copies and to recover using log applied to an Online Consistent Copy, so you should make Online Consistent Copies and use them for recovery only when doing so adds value to your process, such as when there is a requirement for migration or a recovery to a copy.

You can make Online Consistent Copies on either snappable or nonsnappable storage. However, Online Consistent Copies generally perform better when they are on snappable storage. If consistent copies are on nonsnappable storage, you need to specify the DATAMVR installation option (page 552) or specify the DATAMVR option on the OPTIONS statement (page 131) to override the installation option. The DATAMVR option provides XBM with the name of the program to use to copy a data set.

If you request a RECOVER TOCOPY using an Online Consistent Copy that is not available, the fallback recovery is a TIMESTAMP recovery.

NOTE
If you perform a RECOVER TOCOPY using an Online Consistent Copy, you should make a new copy after the recovery completes. After the RECOVER TOCOPY, you will not be able to use copies made before the START_RBA of the Online Consistent Copy.

Solving other problems with RECOVER PLUS

You can use RECOVER PLUS to perform the following tasks:

- recovery from a dropped table space
- migration of data to another subsystem
- creation of a shadow (parallel) space on the same subsystem
- OBID translation
- specification of input copies
- override of log ranges
- conversion of database objects to new definitions with minimal outage

NOTE
Versioned table spaces present special problems during drop recovery or data migration. (See “Handling DB2 versioning information” on page 469.)
# Operational considerations

This chapter presents the following topics:

- Operating environment .......................................................... 78
- DB2 support ........................................................................ 78
- System requirements ............................................................... 78
- Setting the MEMLIMIT parameter ........................................ 79
- Software requirements ........................................................... 80
- Authorization ....................................................................... 80
- DB2 authority ...................................................................... 80
- System authority ................................................................. 81
- APF authority ...................................................................... 82
- Access control authorization exit ......................................... 82
- Table space and index space status ..................................... 83
  - Initial table space and index space status ......................... 84
  - Restoring initial status ....................................................... 86
- Serialization and concurrency issues ................................. 86
  - Concurrency with IBM DB2 utilities ................................. 86
  - Concurrency with other BMC utilities for DB2 ................. 87
  - Concurrency with other RECOVER PLUS executions ......... 89
- Recovery using a non-DB2 backup ..................................... 90
- Recovery using cumulative incremental image copies ......... 90
- Checkpoints for RECOVER PLUS restart ......................... 91
  - Checkpoint installation options .................................... 91
  - Checkpoint override parameter .................................... 92
- Storage group-defined data sets ......................................... 92
  - Primary space allocation ................................................. 93
  - Secondary extents .......................................................... 94
Operating environment

RECOVER PLUS requires the operating environment described in this section.

DB2 support

This version of RECOVER PLUS supports IBM-supported versions of DB2.

For a list of DB2 Version 10 features that RECOVER PLUS supports, see “Version 10.1.00 April 2011” on page 26.

System requirements

This version of RECOVER PLUS supports IBM-supported versions of z/OS that are active on all systems in the sysplex.

Following are some considerations based on operating system level:

- If you are running z/OS Version 1.7 or later, you can use RECOVER PLUS to recover table spaces and index spaces by using large format (which can have more than 64 K tracks) archive logs and image copies.

- If you are running z/OS Version 1.10 or later, RECOVER PLUS supports Extended Address Volumes (EAVs) for DB2 spaces, Instant Snapshot copies, active logs, and the BSDS.

- If you are running z/OS Version 1.11 or later, RECOVER PLUS supports archive logs and sequential image copy data sets in the cylinder-managed portion of EAVs.

**NOTE**

You cannot use an image copy made to the cylinder-managed portion of an extended address volume (EAV) under z/OS Version 1.11 on z/OS Version 1.10 because z/OS Version 1.10 does not support sequential data sets in the cylinder-managed portion of an EAV.

- IDCAMS and DSNUTILB must be available in the system LPA, LINKLIST, JOBLIB, or STEPLIB.
Setting the MEMLIMIT parameter

The following products and components require above-the-bar memory and might abend if sufficient memory is not available:

- ALTER
- BMCSORT
- CATALOG MANAGER
- CHANGE MANAGER
- CHECK PLUS
- COPY PLUS
- DASD MANAGER PLUS
- High-speed Apply Engine
- LOADPLUS
- Log Master
- RECOVER PLUS
- RECOVERY MANAGER
- REORG PLUS
- UNLOAD PLUS

In z/OS versions before 1.10, the default value for the System Management Facility (SMF) MEMLIMIT parameter is 0; a value of 0 means that no address space can use virtual storage above the bar. In z/OS Version 1.10 and later, the default value is 2 GB.

For most jobs, BMC recommends a value of at least 1 GB for the MEMLIMIT parameter. However, if you are operating on LOB or XML data, BMC recommends a value of at least 32 GB.

This value is set in member SMFPRMxx in SYS1.PARMLIB. Use any of the following methods if you need to override the default value:

- Specify the MEMLIMIT parameter in the JCL.
- Specify REGION=0M in the JCL.
- Use the SMF IEFUSI exit.
Software requirements

This version of RECOVER PLUS has the following requirements for additional IBM® or BMC software:

- You must have a minimum of version 10.1.00 of the BMC DB2 Solution Common Code (SCC) installed.

- If you want to offload eligible processing to a zIIP, you must have installed a minimum of version 5.6 with PTF BPE0313 of either XBM or SUF.

If you use the XBMID option to specify a particular XBM subsystem, that subsystem must be at this maintenance level. If you do not specify a particular XBM subsystem and ZIIP ENABLED is in effect, COPY PLUS searches for an XBM subsystem at this level.

- To use any features that invoke DSNUTILB, you must have installed the IBM DB2 RECOVER utility.

Authorization

Using RECOVER PLUS requires that you have authorization within DB2 and (in some cases) through your system security package (such as RACF) that is sufficient to access DB2 resources and perform the tasks accomplished during RECOVER PLUS processing.

DB2 authority

To run RECOVER PLUS, you must have one of the following authorizations:

- installation SYSADM or SYSADM authority

- sufficient authority to run the RECOVER PLUS application plan, and one of the following authorizations:
  — SYSCTRL or system DBADM
  — DBADM or DBCTRL authority for the databases containing the named table spaces
  — RECOVERDB, DISPLAYDB, STARTDB, and STOPDB authority for the databases containing the named table spaces and index spaces
— DATAACCESS, DISPLAYDB, STARTDB, and STOPDB authority for the databases containing the named tables spaces and index spaces

— IMAGCOPY and DISPLAYDB authority to execute OUTCOPY only

Because RECOVER PLUS uses the BMC dynamic bind technology, the OWNER of the plan must be authorized to EXECUTE each package in the plan at the time dynamic bind is performed. If you do not modify the OWNER of the RECOVER PLUS plan specified during installation, you should not need to be concerned with this requirement.

### System authority

Because RECOVER PLUS does not run as part of the DB2 subsystem, you must have system authority similar to that of DB2 to use RECOVER PLUS.

If the underlying data sets of a table space or index space are RACF or similarly protected, you must have sufficient authority to access and modify the data set. If a table or index space is STOGROUP-defined and the corresponding ICF catalog is RACF or similarly protected, you must also have sufficient authority to access and update the operating system catalog. The minimum levels of authority shown in Table 1 are required when you use the following settings:

- The installation option OPNDB2ID is set to NO.
- OPNDB2ID is set to YES and a security system other than RACF is used.

### Table 1  System authority requirements

<table>
<thead>
<tr>
<th>Table or index space definition</th>
<th>Minimum levels of authority required to access and update data sets</th>
<th>Minimum levels of authority required to access and update the operating system catalog</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCAT</td>
<td>CONTROL</td>
<td>none required</td>
</tr>
<tr>
<td>STOGROUP</td>
<td>ALTER or CONTROL</td>
<td>UPDATE (if data set authority is ALTER) or ALTER (if data set authority is CONTROL)</td>
</tr>
</tbody>
</table>

If active logs will be read and OPNDB2ID = NO, the ID running the job needs ALTER authority.

If OPNDB2ID is set to YES and RACF is used, these authorities are not required; in this case the RACF ID for DB2 is used when opening the DB2 data sets or catalog. For more information about the OPNDB2ID installation option, see “OPNDB2ID=YES” on page 539.
APF authority

If DB2 is specified in the RACF started procedures table (ICHRIN03) as a privileged or trusted task and no user ID is associated with the DB2 address space, you cannot use OPNDB2ID to allow RECOVER PLUS to access the DB2 data sets. In this case, the user running RECOVER PLUS must have RACF authority to access the data sets needed for recovery.

The RECOVER PLUS option OPNDB2ID works under data sharing only if all RACF IDs for the members of a group are the same. Authorizations for the bootstrap and log data sets must also be the same.

**NOTE**

You may also use CA ACF2 security or CA Top Secret security from Computer Associates when verifying utility authorizations with RECOVER PLUS if you set the installation option ACFORTSS to YES (which is the default). For more information, see “ACFORTSS=YES” on page 552.

APF authority

RECOVER PLUS uses system services that require APF authorization. Accordingly, RECOVER PLUS must reside in an APF-authorized library.

All load modules loaded by RECOVER PLUS must be authorized and must reside in APF-authorized libraries, as follows:

- IDCAMS
- DSNUTILB
- Data Facility Product (DFP) module IGWASYS (which generally resides in SYS1.CSSLIB)

For DFP Version 3.2 or earlier, this module is called IGWAQSMS.

Access control authorization exit

RECOVER PLUS attempts to use the access control authorization exit that is provided by IBM. If you decide to use the access control authorization exit, you can control access via your selected security package. If you elect not to use the access control authorization exit, RECOVER PLUS works as outlined in “Authorization” on page 80.

For more information, see Appendix A, “Writing exit routines,” in the *DB2 for z/OS Administration Guide*. 

82 RECOVER PLUS for DB2 Reference Manual
Table space and index space status

If the output from a table space or index space recovery is to a DB2 data set, the initial status of the target space must be acceptable to RECOVER PLUS before a recovery can start.

However, if SIMULATE YES is specified or INDEPENDENT OUTSPACE is specified to direct the recovery output of a table space to a non-DB2 data set, the initial status of the space is not a factor in the recovery and is ignored by RECOVER PLUS. Also, no status changes are made to the space. But if you do not specify INDEPENDENT INTABLESPACE when you use the INDEPENDENT OUTSPACE option with REBUILD INDEX, the table space is stopped and started as READ ONLY (RO). In this case, the initial table space status is a factor.

If the recovery is to a DB2 data set and the initial status of the target table space or index space is not acceptable, RECOVER PLUS terminates. If the status is acceptable, RECOVER PLUS stops all of the spaces during the ANALYZE phase to ensure exclusive use of the spaces during the recovery.

RECOVER PLUS sets the appropriate pending status (RECP or RBPD) for each space in anticipation of processing the space. When the recovery of the space is completed successfully, the pending status is turned off. If the job fails, a space may be left in a pending status even though it was never processed.

If you set the RDB2STAT parameter to NO, RECOVER PLUS will leave the spaces stopped to ensure that you take some action before allowing the spaces to be used. For example, you may want to ensure that all related spaces have been recovered to the same point, or you may want to take image copies for future recovery if copies were not taken during processing.

If RECOVER PLUS determines that the spaces can be used when a recovery is complete, the initial status is restored if the RDB2STAT parameter is set to YES. (You can also use RDB2STAT(RW) or RDB2STAT(RO) on the EXEC statement as described on page 274 to set the space in RW or RO status at the end of the run.)

If a table space status is copy pending before a recovery and an output image copy is successfully registered, RECOVER PLUS removes the COPY status. RECOVER PLUS does not remove the COPY status for an image copy that was created with the OUTCOPY ONLY option.

If REBUILD INDEX or RECOVER BUILDINDEX of a nonpartitioned index space on a partitioned space fails during the BUILD phase, RECOVER PLUS sets page set recover pending (PSRCP) or page set rebuild pending (PSRBD) status on the index space.
Initial table space and index space status

Table 2 shows the action taken by RECOVER PLUS for acceptable and unacceptable initial DB2 table space status.

<table>
<thead>
<tr>
<th>RECOVER PLUS command</th>
<th>Action taken by RECOVER PLUS for DB2 table spaces for acceptable initial status.a</th>
<th>Action taken by RECOVER PLUS for DB2 table spaces for unacceptable initial status.a</th>
<th>Unacceptable initial table space status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER TABLESPACE (with or without index recovery or unload keys) without BACKOUT option</td>
<td>Issues the DB2 command STOP AT (COMMIT) for the table space</td>
<td>RECOVER PLUS terminates</td>
<td>RESTART UTUT UTRW UTRO AREST RESP REFPb</td>
</tr>
<tr>
<td>RECOVER TABLESPACE (with or without index recovery or unload keys) with BACKOUT option</td>
<td>Issues the DB2 command STOP AT (COMMIT) for the table space</td>
<td>RECOVER PLUS terminates</td>
<td>RECP LPL GRECP DEFER WEPR REFP UTUT UTRW UTRO</td>
</tr>
<tr>
<td>REBUILD INDEX or RECOVER UNLOADKEYS (without table space recovery)</td>
<td>Issues the DB2 command STOP AT (COMMIT) for the table space and restart in ACCESS(RO)</td>
<td>RECOVER PLUS terminates</td>
<td>RESTART UTUT UTRW RECP DEFER LPL GRECP AREST RESTP WEPR REFP</td>
</tr>
</tbody>
</table>

a  When the recovery is to a non-DB2 data set, the initial status of a table space is ignored and no status changes are made.

b  Acceptable if the recovery is to a point in time.
Table 3 shows the action taken by RECOVER PLUS for acceptable and unacceptable initial DB2 index space status.

**Table 3** RECOVER PLUS actions for access to DB2 index spaces

<table>
<thead>
<tr>
<th>RECOVER PLUS command</th>
<th>Action taken by RECOVER PLUS for DB2 index spaces for acceptable initial status&lt;sup&gt;a&lt;/sup&gt;</th>
<th>Action taken by RECOVER PLUS for DB2 index spaces for unacceptable initial status&lt;sup&gt;a&lt;/sup&gt;</th>
<th>Unacceptable initial index space status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX or RECOVER INDEXSPACE without BACKOUT option</td>
<td>Issues the DB2 command STOP AT (COMMIT) for the index space</td>
<td>RECOVER PLUS terminates</td>
<td>RESTART UTUT UTRW UTRER REPTP REFP&lt;sup&gt;b&lt;/sup&gt;</td>
</tr>
<tr>
<td>RECOVER INDEX or RECOVER INDEXSPACE with BACKOUT option</td>
<td>Issues the DB2 command STOP AT (COMMIT) for the index space</td>
<td>RECOVER PLUS terminates</td>
<td>RECP RESTART UTUT UTRW UTRER RBDP PSRCPSRBD WEP REFP</td>
</tr>
<tr>
<td>REBUILD INDEX or RECOVER BUILDINDEX</td>
<td>Issues the DB2 command STOP AT (COMMIT) for the index space</td>
<td>RECOVER PLUS terminates</td>
<td>RESTART UTUT UTRW UTRER REPTP</td>
</tr>
<tr>
<td>RECOVER UNLOADKEYS</td>
<td>No action is required, any status is acceptable</td>
<td>No action is required, any status is acceptable</td>
<td>N/A</td>
</tr>
</tbody>
</table>

<sup>a</sup> When the recovery is to a non-DB2 data set, the initial status of a table space is ignored and no status changes are made.

<sup>b</sup> Acceptable if the recovery is to a point in time.

When the initial status is acceptable, RECOVER PLUS issues DB2 commands that give control of the space to RECOVER PLUS and allow the recovery to start. When the initial status is not acceptable, RECOVER PLUS terminates. However, if the output from an index or table space recovery is to be directed to a non-DB2 data set, RECOVER PLUS does not issue any DB2 commands.
**Restoring initial status**

RECOVER PLUS provides the RDB2STAT utility parameter, which determines whether DB2 objects should be returned to their original status when a recovery job is complete. If you specify RDB2STAT(YES) (or default to the RDB2STAT installation option when it is set to YES), RECOVER PLUS handles the space as follows:

- starts the space with ACCESS(RW) status if the space was in RW status initially
- starts the space with ACCESS(RO) if the space was in RO status
- will not start the space if the space was in STOP, STOPE, LSTOP, or STOPP status
- starts the space with START ACCESS (FORCE) if the space was in DEFER, LPL status, GRECP status, or internal ERROR RANGE status

You may also use RDB2STAT(RW) and RDB2STAT(RO) on the EXEC statement to request that DB2 objects be restored to a status of RW or RO, regardless of their initial status. For a description of the RDB2STAT parameter, see “RDB2STAT override parameter (rdb2Stat)” on page 274.

If you do not specify RDB2STAT, RECOVER PLUS defaults to the value that is specified for the RDB2STAT installation option.

**Serialization and concurrency issues**

RECOVER PLUS allows you to run the utility concurrently with other RECOVER PLUS runs, with other utilities, and with applications that access target table spaces.

**Concurrency with IBM DB2 utilities**

RECOVER PLUS uses a DB2 command interface to obtain the initial DB2 status of each space involved in the recovery and determine whether an IBM DB2 utility is already operating against that space. If the DB2 status of a space indicates that an IBM DB2 utility is operating against the space, RECOVER PLUS terminates. Otherwise, RECOVER PLUS issues DB2 commands to stop all of the spaces to ensure exclusive use of the space and prevent any IBM utility from operating against the space until the recovery is complete. STOP and START commands are issued by partition as appropriate. For more information, see “Table space and index space status” on page 83.
Concurrency with other BMC utilities for DB2

All BMC utility products use the BMCUTIL table to control the use of utility IDs, which identify executions of BMC utilities. Only one utility with a specific utility ID can run at one time, and RECOVER PLUS terminates when it is started for a space against which another BMC utility is already running. For more information about this table, see “BMCUTIL table” on page 580.

BMC utility products use the BMCSYNC table to coordinate access to DB2 objects. DB2 objects that participate in a BMC utility job are registered in the BMCSYNC table. When each object is registered, the registering utility assigns a share level to control access to that object from other BMC utilities. For partitioned DB2 spaces, registration is performed at the partition level. For more information about this table, see “BMCSYNC table” on page 573.

The BMCSYNC table allows multiple BMC utilities (or multiple instances of a single utility) to operate concurrently on different partitions of a DB2 space if no nonpartitioning indexes are involved. In addition, some BMC utilities can operate concurrently on the same object or partition. For information about which products can operate concurrently, see Table 4.

The “Access level” column in Table 4 refers to the value of the SHRLEVEL column in the BMCSYNC table. The level can be one of the following values:

- S indicates shared access. Any other utility that registers with shared access (S) can run against the object.

- X indicates exclusive access. No other utility can run against the object.

- A blank value indicates that no status is requested and any other utility can run against the object.

Table 4  Running BMC products concurrently (part 1 of 2)

<table>
<thead>
<tr>
<th>Product</th>
<th>Access level</th>
<th>Additional information</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHECK PLUS</td>
<td>S</td>
<td>none</td>
</tr>
</tbody>
</table>
| COPY PLUS                      | S or blank   | ■ If you specify COPY IMAGECOPY, COPY PLUS registers the object with no access status (blank).  
■ In all other cases, COPY PLUS registers the object with shared access (S). |
| DASD MANAGER PLUS (BMCSTATS)   | S            | none                                                                                   |
Concurrency with other BMC utilities for DB2

### Table 4 Running BMC products concurrently (part 2 of 2)

<table>
<thead>
<tr>
<th>Product</th>
<th>Access level</th>
<th>Additional information</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOADPLUS</td>
<td>X</td>
<td>If you specify PART, LOADPLUS registers only the specified partitions with exclusive access (X). If no nonpartitioned indexes exist on the table space, you can run other utilities on different partitions concurrently with this job.</td>
</tr>
<tr>
<td>RECOVER PLUS</td>
<td>X, S, or blank</td>
<td>Under the following conditions, RECOVER PLUS registers an object with shared access (S):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— If an index is being rebuilt, the table space for that index is registered with shared access if that table space is not also recovered in the same job.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— A table space partition is registered with shared access if the keys for that partition are unloaded with a RECOVER UNLOADKEYS operation.</td>
</tr>
<tr>
<td>RECOVERY MANAGER</td>
<td>S</td>
<td>none</td>
</tr>
<tr>
<td>REORG PLUS</td>
<td>X</td>
<td>If you specify PART, REORG PLUS registers only the specified partitions with exclusive access (X). If no nonpartitioned indexes exist on the table space, you can run other utilities on different partitions concurrently with this job.</td>
</tr>
<tr>
<td>UNLOAD PLUS</td>
<td>S</td>
<td>none</td>
</tr>
</tbody>
</table>
Concurrency with other RECOVER PLUS executions

RECOVER PLUS allows a high degree of concurrency between RECOVER PLUS jobs. You can recover different partitions of the same table space or index space concurrently in different RECOVER PLUS jobs. However, the following scenarios are not allowed in RECOVER PLUS:

- submit two RECOVER PLUS jobs concurrently to recover all of the data sets of a space or the same data set of a space
- recover a table space partition and unload keys from the same partition in two different concurrent jobs

The utility issues a warning if you run more than one job that specifies RECOVER UNLOADKEYS for the same index and from the same partition or partitions of a table space.

Concurrency with RECOVER UNLOADKEYS and RECOVER BUILDINDEX

A special case of concurrency is one or more RECOVER UNLOADKEYS runs followed by a RECOVER BUILDINDEX run. Because these operations can be run in separate jobs, updates to the table space must be prevented between the completion of RECOVER UNLOADKEYS and the beginning of RECOVER BUILDINDEX to guarantee the integrity of the data. For this reason, RECOVER UNLOADKEYS leaves the table space in RO (read only) status. RECOVER PLUS also adds a row to the BMCSYNC table for each partition from which keys have been unloaded; the utility ID is blank and SHRLEVEL is “S”. This status prevents any other RECOVER PLUS run from recovering the table space before the row is removed by the RECOVER BUILDINDEX run.

In the same way, the RECOVER UNLOADKEYS run adds a row to the BMCSYNC table for each nonpartitioned index unloaded; the utility ID is blank and the BMCSYNC row is shared among all of the RECOVER UNLOADKEYS jobs for the same index. The RECOVER BUILDINDEX job deletes the BMCSYNC rows with blank utility IDs that were left by the corresponding RECOVER UNLOADKEYS jobs. The RECOVER BUILDINDEX job also checks the row for each index first to verify that all partitions have been unloaded, and issues a warning message if the row is not found. The job terminates with an error if it finds the row with indications that not all partitions have been unloaded.

For specific examples of concurrent RECOVER PLUS jobs, see “Using multiple commands” on page 261.
Recovery using a non-DB2 backup

RECOVER PLUS allows you to recover a space without the use of a DB2 image copy. You can accomplish this task by restoring the space from a non-DB2 backup and then by running a RECOVER PLUS job that uses either the LOGAPPLY ONLY or LOGONLY option. An entire disk pack or a single space can be recovered in this manner.

When you use the LOGAPPLY ONLY option, you must specify FROMLOGPOINT. You must select this log point with care to ensure that it represents a point prior to the non-DB2 backup at which all buffers were externalized. A good rule is to select a log point that is three checkpoints prior to the time of the backup. A quiesce point prior to the time of the backup is also an acceptable log point. Unpredictable results may occur if the selected value of FROMRBA or FROMLOGPOINT is in a time when the space is in an exceptional state such as error range status or deferred status.

When a value for FROMLOGPOINT is established, the space or disk pack should be restored from the non-DB2 backup. The restoration is followed by the RECOVER PLUS run that uses LOGAPPLY ONLY and the selected value of FROMLOGPOINT.

The LOGONLY option uses the log point that is recorded in the space as the starting log point.

Recovery using cumulative incremental image copies

By using the BMC COPY PLUS for DB2 product, you can merge incremental image copies while retaining the previous incremental copies in SYSIBM.SYSCOPY (BMCXCOPY for incremental index copies) by specifying CUMULATIVE YES KEEP YES. If you specify these options for an incremental copy, the previous copy is registered as ICTYPE=i. This copy is referred to as a cumulative copy. For incremental copies of table spaces, to make this copy available for DB2 RECOVER you must run the COPY PLUS RECALL job to set ICTYPE=I.

If a space is recovered to the current point, RECOVER PLUS ignores incremental copies with ICTYPE=i. If a space is recovered to a prior point in time, RECOVER PLUS recognizes any incremental copies with ICTYPE=i that would be useful for the recovery.
For example, if you are recovering to a cumulative incremental copy, RECOVER PLUS treats the cumulative incremental copy as a true copy (ICTYPE=I). If you are recovering to a specific log point and a cumulative incremental copy exists prior to that log point, RECOVER PLUS merges the cumulative incremental copy with the full copy before applying the log. Fallback processing also recognizes cumulative incremental copies and uses those copies as necessary.

For more information about the CUMULATIVE, KEEP, and RECALL options, see the COPY PLUS for DB2 Reference Manual.

Checkpoints for RECOVER PLUS restart

You can facilitate restarting a RECOVER PLUS job that fails to complete successfully by specifying the checkpoints to be taken in the original run. Processing phases are not re-executed during the restarted job if checkpoints were taken. The end of a phase is recorded in the BMCSYNC table, and a job can be restarted from the last recorded checkpoint.

Two installation options, CHECKPT and CHECKINT, determine when checkpoints are taken for RECOVER PLUS jobs. In addition, you can override the CHECKINT installation option at run time by using the RECOVER PLUS OPTIONS command with the CHECKINT keyword.

Checkpoint installation options

You can specify the CHECKPT installation option to take no checkpoints, or to take checkpoints at the end of each phase. When you specify checkpoints to be taken, you can prevent taking unnecessary checkpoints by specifying a value for the CHECKINT (checkpoint interval) installation option, which specifies the minimum number of minutes that must elapse before the next checkpoint is taken. To decide on values for CHECKPT and CHECKINT, you must balance the cost of taking checkpoints against the time lost redoing work when a RECOVER PLUS execution fails and must be restarted. For more information about these installation options, see Appendix A, “RECOVER PLUS installation options.”
Checkpoint override parameter

At run time, you can override the value for the CHECKPT installation option by specifying a value for the checkpoint override parameter in your RECOVER PLUS JCL. Use the following guidelines:

- Specify CHECKPT(NO) (no check points) for short recoveries that you do not mind rerunning when necessary.

- Specify CHECKPT(PHASE) (checkpoints at the end of each phase) for longer running jobs when it becomes costly to rerun the entire job.

As with the CHECKPT installation option, the setting of CHECKINT controls the frequency of the taking of checkpoints when you use the checkpoint override parameter.

For more information about the checkpoint override parameter, see Chapter 4, “Building and executing RECOVER PLUS jobs.”

Storage group-defined data sets

For a storage group-defined space, when RECOVER PLUS defines the underlying data set, RECOVER PLUS uses the following algorithm:

1. Obtain volume information for the storage group definition from SYSIBM.SYSVOLUMES in the DB2 system catalog.

2. If the data set is currently cataloged, obtain the volume information from the operating system catalog entry. Arrange to favor those volumes if they are still part of the STOGROUP definition.

3. Determine whether the data set is SMS managed by calling IGWAQSMS (DFP 3.2) or IGWASYS (DFP 3.3 or later).

4. If the data set exists, delete it (unless REDEFINE NO is specified). If it is not on the volume indicated by the operating system catalog entry, delete the data set with the NOSCRATCH option of IDCAMS. If REDEFINE YES NOSCRATCH is specified, no attempt to scratch the data set is made; it is simply uncataloged with the NOSCRATCH option of IDCAMS.
5. Define the data set. The following pseudo code shows which volume is selected for the initial definition of the data set and any subsequent extend requests that require a new volume. If a define or extend request fails, RECOVER PLUS tries the algorithm again. Volumes on which an extend request fails are internally marked as tried.

```
IF THE STOGROUP DEFINITION INCLUDES SPECIFIC VOLSERS AND ANY VOLSER UNTRIED
    DEFINE/EXTEND USING THE NEXT UNTRIED VOLSER
ELSE
    IF THE STOGROUP DEFINITION INCLUDES '*' AND THE DATA SET IS SMS MANAGED
        DEFINE/EXTEND USING '*'
    ELSE
        DEFINE/EXTEND USING AN ORIGINAL VOLSER THAT IS UNTRIED
```

**NOTE**
Specific volume serial numbers are used in the order in which they are returned by SQL from SYIBM.SYSVOLUMES. For more information about how to change the volume ordering, please see the discussion of the STOGROUP USEORDER option in “STOGROUP specification description” on page 117.

If a nonpartitioned space is in multiple data sets, unused data sets are removed even with the REDEFINE option set to NO.

If the extend request requires a new data set or volume, it will be defined even with the REDEFINE option set to NO.

If an Instant Snapshot copy is used to recover the object, REDEFINE YES has no effect and the data sets are processed as if REDEFINE NO had been specified.

**Primary space allocation**

For DB2 Version 8 or later, if the PRIQTY value is specified, it is used for primary space allocation. If PRIQTY is not specified, the DSNZPARM TSQTY value is used. If TSQTY is set to 0, then 1 cylinder is the default primary space allocation.
Secondary extents

For DB2 Version 8 or later, the SECQTY value is used for secondary extents if DSNZPARM MGEXTSZ is set to NO. If MGEXTSZ is set to YES or SECQTY is not specified, RECOVER PLUS uses a sliding-scale calculation for secondary extents, similar to the method DB2 uses.

NOTE

For more information about the sliding-scale calculation, see the DB2 for z/OS SQL Reference.
RECOVER PLUS syntax

This chapter illustrates command syntax and describes each command option.

**NOTE**
For quick reference, the syntax diagrams are duplicated without any descriptive text in Appendix E, “RECOVER PLUS syntax diagrams.”

Overview ......................................................... 96
OPTIONS .......................................................... 96
OUTPUT ......................................................... 97
RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE .................. 97
REBUILD INDEX .................................................. 98
RECOVER UNLOADKEYS ...................................... 98
RECOVER BUILDINDEX ........................................ 99
LOGSCAN ....................................................... 99
Adding comments ............................................. 99
Use of long names ........................................... 100
Support for Unicode names ................................. 101
Alphabetical listing of RECOVER PLUS options ......................... 102
OPTIONS .......................................................... 109
OPTIONS syntax ............................................... 109
OPTIONS option descriptions ................................ 112
OUTPUT ......................................................... 145
OUTPUT syntax ............................................... 145
OUTPUT command dependencies and prohibitions .................... 146
OUTPUT option descriptions ................................ 147
RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE .................. 155
RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE
  syntax .......................................................... 155
RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX
  command dependencies and prohibitions ........................ 164
RECOVER INDEX and RECOVER INDEXSPACE command dependencies and
  prohibitions .................................................. 165
Overview

This chapter provides brief descriptions of the purpose and usage of the commands that you can use with RECOVER PLUS.

For information about using multiple RECOVER PLUS commands, see “Using multiple commands” on page 261.

OPTIONS

The OPTIONS command specifies options that are set globally for use in a RECOVER PLUS job. You can have more than one OPTIONS command statement in a job but a particular specification can only be defined one time. You must place the OPTIONS command statement or statements before any other RECOVER PLUS command statement in your job.

For more information, see “OPTIONS” on page 109.
OUTPUT

The OUTPUT command specifies options that you can code for the dynamic allocation of output data sets. Options are available for disk data sets and for tape data sets.

For more information, see “OUTPUT” on page 145.

RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE

The RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE commands specify one or more table spaces or indexes, or table space or index data set, for recovery.

You can recover a single table space, index, or data set to a specific image copy, to a specific log point, or to the current state (by using all of the log records that apply).

Multiple table spaces, indexes, or data sets can be recovered to a specific log point or to the current state. You can have as many RECOVER TABLESPACE, RECOVER INDEX, or RECOVER INDEXSPACE command statements in a job step as necessary. However, unlike the IBM DB2 RECOVER utility, RECOVER PLUS examines all of the command statements in the job step before processing starts so that all log processing can be combined and activities can be optimally scheduled.

NOTE
If you request simulation mode with the SIMULATE YES option, which requires a valid Recovery Management for DB2 solution password, you must use SIMRCVR TABLESPACE, SIMRCVR INDEX, and SIMRCVR INDEXSPACE instead of RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE.

For more information, see “RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE” on page 155.

NOTE
This book assumes that RECOVER INDEX uses copies, logs, or both. However, RECOVER PLUS can interpret the RECOVER INDEX syntax as REBUILD INDEX if the option INDEXLOG is set to NO (the default). INDEXLOG AUTO could have the same effect in some circumstances. INDEXLOG is set by using the INDEXLOG installation option or the INDEXLOG option on the OPTIONS command (see “INDEXLOG=NO” on page 541 and “INDEXLOG” on page 126).
The REBUILD INDEX command specifies that one or more indexes on tables in a single table space are to be rebuilt from the data in the table space. If the index is a partitioned index on a partitioned table space and you do not specify a partition, RECOVER PLUS rebuilds all partitions of a partitioned index. You can specify a single partition of the index.

You can include multiple REBUILD INDEX command statements in a single RECOVER PLUS execution, along with RECOVER TABLESPACE and RECOVER UNLOADKEYS command statements. All command statements are examined and processed as efficiently as possible.

For more information, see “REBUILD INDEX” on page 212.

**NOTE**
If you request simulation mode with the SIMULATE YES option, which requires a valid Recovery Management for DB2 solution password, you must use SIMRBLD INDEX rather than REBUILD INDEX.

**RECOVER UNLOADKEYS**

The RECOVER UNLOADKEYS and RECOVER BUILDINDEX commands work together but in separate executions. RECOVER UNLOADKEYS extracts from the table space the index keys that RECOVER BUILDINDEX uses to build nonpartitioned indexes on a partitioned table space.

RECOVER UNLOADKEYS specifies nonpartitioned indexes on a partitioned table space and specifies one partition of the table space from which the keys for those indexes are extracted. The keys are written to data sets which are specified in a RECOVER BUILDINDEX command statement in a subsequent job. The keys within these data sets are sorted during the RECOVER UNLOADKEYS processing.

To extract keys from multiple partitions, you use multiple RECOVER UNLOADKEYS command statements. You can include these with RECOVER TABLESPACE and REBUILD INDEX command statements.

For more information, see “RECOVER UNLOADKEYS” on page 228.

**NOTE**
If you request simulation mode with the SIMULATE YES option, which requires a valid Recovery Management for DB2 solution password, you must use SIMRRCVR UNLOADKEYS rather than RECOVER UNLOADKEYS.
RECOVER BUILDINDEX

The RECOVER BUILDINDEX command specifies the rebuilding of one or more nonpartitioned indexes on a partitioned table space by using data sets that are created in earlier RECOVER UNLOADKEYS runs. You can include only one RECOVER BUILDINDEX command statement in the RECOVER PLUS execution; no other RECOVER PLUS command statements are allowed.

For more information, see “RECOVER BUILDINDEX” on page 237.

**NOTE**
If you request simulation mode with the SIMULATE YES option, which requires a valid Recovery Management for DB2 solution password, you must use SIMRCVR BUILDINDEX rather than RECOVER BUILDINDEX.

LOGSCAN

The LOGSCAN command provides information about the number and size of log records required for recovery. You can use this information to size the log sort. RECOVER PLUS prints a report describing the log records for each object that is named in the LOGSCAN.

For more information, see “LOGSCAN” on page 244.

Adding comments

You can code comments in RECOVER PLUS command statements by bracketing the comment with /* at the beginning and */ at the end. For example:

```
//SYSIN DD*
/*Recover production table space*/
RECOVER TABLESPACE PRODDB.PRODTS
   TORBA LASTQUIESCE /*prior to last batch update*/
```

**NOTE**
When coding /*, avoid column 1 because this string could terminate prematurely your SYSIN.
You can also code comments by preceding information with a double hyphen (--). A comment that is started with the hyphens runs to the end of the line. You can place the double hyphen in column 1 through column 70. Do not break the double hyphen across a line. An example follows:

```
//SYSIN DD*
-- Recover production table space
RECOVER TABLESPACE PRODDB.PRODTS
   TORBA LASTQUIESCE --prior to last batch update
```

All characters inside comments, both those specified with /* and those specified with a double hyphen, are ignored.

## Use of long names

RECOVER PLUS supports long names up to 128 bytes in length for the following identifiers:

- table names
- index names
- creator names
- STOGROUP names

The only long names that you can use in the SYSIN file are creator names and index names. RECOVER PLUS reads the long names from SYSIN, parses them, saves them in control blocks, and displays long names in output messages. Output messages can contain long creator, index, and table names, which may cause messages to require multiple lines.

In SYSIN, long names must be in columns 1 to 72. Columns 73 to 80 are ignored. You can split long names in SYSIN across lines. If a name is split across lines, the name must continue to column 72 with no embedded spaces, and the remainder of the name must start in column 1 on the next line.
Support for Unicode names

RECOVER PLUS provides support for Unicode names as follows:

- Unicode is not supported in the SYSIN file. However, RECOVER PLUS can process spaces that contain tables or indexes with Unicode names. (Spaces cannot contain Unicode names in DB2.) RECOVER PLUS processes spaces with STOGROUP names, index names, and index creator names in Unicode.

RECOVER PLUS commands with wild cards do not include objects that match the pattern but contain Unicode characters that are not translatable to EBCDIC in the wild card position. This result is because SYSIN is EBCDIC and wild card processing is done in EBCDIC.

- In output files, RECOVER PLUS displays Unicode names for index names (not index space names) or table names that do not translate to EBCDIC as a UTF-8 (Unicode Transformation Format, 8-bit encoding form) representation in hexadecimal delimited by angle brackets (<>).

Figure 2 shows an example of how RECOVER PLUS represents Unicode in SYSPRINT.

**Figure 2  Unicode representation in the RECOVER PLUS output file**

<table>
<thead>
<tr>
<th>BMC40701I</th>
<th>PHASE:  MERGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40721I</td>
<td>PHASE INITIALIZATION:</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = RDACMC.ACP&lt;E18699&gt;.IN01P1&lt;E18699&gt;</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC40702I</td>
<td>STEP:  MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = RDACMC.ACP&lt;E18699&gt;.IN01P1&lt;E18699&gt;</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
</tbody>
</table>
Alphabetical listing of RECOVER PLUS options

RECOVER PLUS options are listed in Table 5, alphabetized by RECOVER PLUS command, and within the command by option name. The last column contains a page reference to a description for each option.

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGSCAN</td>
<td>244</td>
<td>AUX</td>
<td>250, 254</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BACKOUT</td>
<td>259</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
<td>255</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM</td>
<td>252</td>
</tr>
<tr>
<td>FROMLOGPOINT</td>
<td></td>
<td>□ LASTARCHQ</td>
<td>250</td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTCOPY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTQUIESCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTSHUTDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ X’logPoint’</td>
<td></td>
</tr>
<tr>
<td>FROMRBA</td>
<td></td>
<td>□ LASTARCHQ</td>
<td>250</td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTCOPY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTQUIESCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTSHUTDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ X’logPoint’</td>
<td></td>
</tr>
<tr>
<td>INDEX</td>
<td></td>
<td>□ for single index specification</td>
<td>252</td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ for multiple index specification</td>
<td>253</td>
</tr>
<tr>
<td>INDEXES (or INDEX)</td>
<td></td>
<td></td>
<td>254</td>
</tr>
<tr>
<td>INDEXSPACE</td>
<td></td>
<td>□ for single index specification</td>
<td>252</td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ for multiple index specification</td>
<td>254</td>
</tr>
<tr>
<td>LOGAPPLY ONLY</td>
<td></td>
<td></td>
<td>260</td>
</tr>
<tr>
<td>LOGONLY</td>
<td></td>
<td></td>
<td>260</td>
</tr>
<tr>
<td>OBJECTSET</td>
<td></td>
<td></td>
<td>255</td>
</tr>
<tr>
<td>TABLESPACE</td>
<td></td>
<td></td>
<td>248</td>
</tr>
<tr>
<td>TOLOGPOINT</td>
<td></td>
<td>□ LASTARCHQ</td>
<td>257</td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTCOMMONQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTQUIESCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(relativeGenerationNumber)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTSHUTDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ X’logPoint’</td>
<td></td>
</tr>
<tr>
<td>TORBA</td>
<td></td>
<td>□ LASTARCHQ</td>
<td>257</td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTCOMMONQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTQUIESCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(relativeGenerationNumber)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ LASTSHUTDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>□ X’logPoint’</td>
<td></td>
</tr>
<tr>
<td>Command name</td>
<td>Page</td>
<td>Command option</td>
<td>Page</td>
</tr>
<tr>
<td>--------------</td>
<td>------</td>
<td>----------------</td>
<td>------</td>
</tr>
<tr>
<td>OPTIONS or OPTION</td>
<td>109</td>
<td>ANALYZE</td>
<td>132</td>
</tr>
<tr>
<td>AUX</td>
<td>122</td>
<td>BACKOUT</td>
<td>124</td>
</tr>
<tr>
<td>CHECKINT</td>
<td>117</td>
<td>DATAMVR</td>
<td>131</td>
</tr>
<tr>
<td>EARLYCAT</td>
<td>112</td>
<td>EARLYRECALL</td>
<td>112</td>
</tr>
<tr>
<td>INDEXLOG</td>
<td>126</td>
<td>IXRECP</td>
<td>128</td>
</tr>
<tr>
<td>KEYSORT</td>
<td>131</td>
<td>KSORTSHARE</td>
<td>139</td>
</tr>
<tr>
<td>LOGPOINT X’logPoint’ or LOGPOINT LOGMARK logMarkName (logMarkGeneration)</td>
<td>143</td>
<td>MAXDRIVES</td>
<td>120</td>
</tr>
<tr>
<td>MAXLOGS</td>
<td>121</td>
<td>MAXLSORT</td>
<td>138</td>
</tr>
<tr>
<td>NOEARLYCAT</td>
<td>112</td>
<td>NOEARLYRECALL</td>
<td>112</td>
</tr>
<tr>
<td>NOSYSLOGRNG</td>
<td>116</td>
<td>NUMREC</td>
<td>115</td>
</tr>
<tr>
<td>ABS</td>
<td></td>
<td>AVGRECSZ</td>
<td></td>
</tr>
<tr>
<td>CALC</td>
<td></td>
<td>EST</td>
<td></td>
</tr>
<tr>
<td>NOEST</td>
<td></td>
<td>ON ERROR ANY CONTINUE</td>
<td>135</td>
</tr>
<tr>
<td>OUTCOPY</td>
<td>121</td>
<td>ASCODED</td>
<td></td>
</tr>
<tr>
<td>BYPART</td>
<td></td>
<td>RECOVERYPOINT</td>
<td>140</td>
</tr>
<tr>
<td>RESINV</td>
<td>119</td>
<td>RESOURCE SELECTION</td>
<td>120</td>
</tr>
<tr>
<td>ACCUMS</td>
<td></td>
<td>COPIES</td>
<td></td>
</tr>
<tr>
<td>LOGS</td>
<td></td>
<td>SIMULATE YES</td>
<td>134</td>
</tr>
<tr>
<td>SMCORE</td>
<td>119</td>
<td>SORTDEVT</td>
<td>131</td>
</tr>
<tr>
<td>for KEYSORT</td>
<td></td>
<td>for LOGSORT</td>
<td>114</td>
</tr>
</tbody>
</table>
### Table 5  RECOVER PLUS command options (part 3 of 7)

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTIONS or OPTION (continued)</td>
<td>109</td>
<td>SORTDIAG</td>
<td>118</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDYN</td>
<td>119</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for KEYSORT</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for LOGSORT</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>STOGRAM... USEORDER</td>
<td>117</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TIMESTAMP</td>
<td>143</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRTCH</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td></td>
<td>URIDDDN</td>
<td>142</td>
</tr>
<tr>
<td></td>
<td></td>
<td>USEACCUM</td>
<td>116</td>
</tr>
<tr>
<td></td>
<td></td>
<td>USEHDROBIDS</td>
<td>123</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WTOR</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XBMID</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZIIP</td>
<td>130</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>145</td>
<td>CATLG</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DATACLAS</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNAMES</td>
<td>147</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EATTR</td>
<td>151</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXPDT</td>
<td>154</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MGMTCLAS</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODELDCB</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RETPD</td>
<td>153</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPACE</td>
<td>151</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STACK</td>
<td>152</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STORCLAS</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRTCH</td>
<td>153</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNIT</td>
<td>148</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNITCNT</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VOLUMNT</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VOLUMNT</td>
<td>151</td>
</tr>
</tbody>
</table>
### Table 5  RECOVER PLUS command options (part 4 of 7)

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>REBUILD INDEX</td>
<td>212</td>
<td>ANALYZE</td>
<td>225</td>
</tr>
<tr>
<td>SIMRBLD INDEX</td>
<td></td>
<td>AUX</td>
<td>219</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
<td>221</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP INTABLESPACE</td>
<td>219</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP OUTSPACE</td>
<td>217</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>for INDEP INTABLESPACE</td>
<td>219</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for INDEP OUTSPACE</td>
<td>217</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSCRATCH</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NUMREC</td>
<td>224</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ABS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CALC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>EST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOEST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOWORKDDN</td>
<td>222</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
<td>216</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PART</td>
<td>220</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDEFINE</td>
<td>226</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REUSE</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT</td>
<td>223</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
<td>223</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
<td>219</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WORKDDN</td>
<td>221</td>
</tr>
<tr>
<td>RECOVER BUILDDINDEX</td>
<td>237</td>
<td>ANALYZE</td>
<td>241</td>
</tr>
<tr>
<td>SIMRCVR BUILDDINDEX</td>
<td></td>
<td>CLONE</td>
<td>241</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP OUTSPACE</td>
<td>239</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
<td>240</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSCRATCH</td>
<td>243</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDEFINE</td>
<td>242</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REUSE</td>
<td>243</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEY</td>
<td>241</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEYDDN</td>
<td>241</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
<td>240</td>
</tr>
</tbody>
</table>
### Table 5  RECOVER PLUS command options (part 5 of 7)

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX</td>
<td>155</td>
<td>ANALYZE</td>
<td>189</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>AUX</td>
<td>169,175</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>BACKOUT</td>
<td>184</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>CLONE</td>
<td>189</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>DBID</td>
<td>194</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>DROPRECOVERY</td>
<td>170</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNAMES</td>
<td>204</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM for INDEX</td>
<td>172</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM for INDEXSPACE</td>
<td>172</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM for TABLESPACE</td>
<td>169</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSSIZE</td>
<td>171</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ENCRYPTED</td>
<td>206</td>
</tr>
<tr>
<td>FROMLOGPOINT</td>
<td></td>
<td>FROMRBA</td>
<td>198</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTARCHQ</td>
<td>198</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTCOPY</td>
<td>198</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTQUIESCE</td>
<td>198</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTSHUTDOWN</td>
<td>198</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X’logPoint’</td>
<td>198</td>
</tr>
<tr>
<td>INDEXES (or INDEX)</td>
<td></td>
<td>INDEXES (or INDEX)</td>
<td>168</td>
</tr>
<tr>
<td>INDEX</td>
<td></td>
<td>INDEX for single index spec</td>
<td>172</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEX for multiple index spec</td>
<td>174</td>
</tr>
<tr>
<td>INDEXSPACE</td>
<td></td>
<td>INDEXSPACE for single index</td>
<td>172</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXSPACE for multiple index</td>
<td>175</td>
</tr>
<tr>
<td>INDEPENDENT SPACE</td>
<td></td>
<td>INDEPENDENT OUTSPACE</td>
<td>178</td>
</tr>
<tr>
<td>INLINE</td>
<td></td>
<td>INLINE</td>
<td>196</td>
</tr>
<tr>
<td>INLOG</td>
<td></td>
<td>INLOG LOGPOINT</td>
<td>197</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INLOG RBA</td>
<td>197</td>
</tr>
<tr>
<td>INSEQNO</td>
<td></td>
<td>INSEQNO</td>
<td>207</td>
</tr>
<tr>
<td>INVOLUME</td>
<td></td>
<td>INVOLUME</td>
<td>207</td>
</tr>
<tr>
<td>LATESTCOPY</td>
<td></td>
<td>LATESTCOPY</td>
<td>186</td>
</tr>
<tr>
<td>(relativeGenerationNumber)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOCALSITE</td>
<td></td>
<td>LOCALSITE</td>
<td>188</td>
</tr>
</tbody>
</table>
### Table 5   RECOVER PLUS command options (part 6 of 7)

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX</td>
<td>155</td>
<td>LOGAPPLY ONLY</td>
<td>188</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>LOGONLY</td>
<td>188</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>LOGPOINT</td>
<td>206</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>LOGSORT</td>
<td>188</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>MODEL</td>
<td>180</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>NOCOPYPEND</td>
<td>197</td>
</tr>
<tr>
<td>(continued)</td>
<td></td>
<td>NOSCRATCH</td>
<td>191</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOWORKDDN</td>
<td>173, 175</td>
</tr>
<tr>
<td>NUMREC</td>
<td></td>
<td>OBID</td>
<td>195</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBIDXLAT</td>
<td>194</td>
</tr>
<tr>
<td>INTERVAL</td>
<td></td>
<td>OBJECTSET</td>
<td>176</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPY</td>
<td>199</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPYDDN</td>
<td>208</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPYDSN</td>
<td>208</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PIECESIZE</td>
<td>205</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSID</td>
<td>195</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RBA</td>
<td>206</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYDDN</td>
<td>209</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYDSN</td>
<td>209</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYSITE</td>
<td>189</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDEFINE</td>
<td>191</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REGISTER</td>
<td>202</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESET</td>
<td>194</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESETRTS</td>
<td>203</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REUSE</td>
<td>192</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SHRLEVEL</td>
<td>206</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SNAPSHOT</td>
<td>196</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPACE</td>
<td>204</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
<td>168</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for table space specification</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>for multiple index specification</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>TOCOPY</td>
<td>185</td>
</tr>
</tbody>
</table>
## Table 5  RECOVER PLUS command options (part 7 of 7)

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX</td>
<td>155</td>
<td>TOLOGPOINT</td>
<td>181</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>▪ LASTARCHQ</td>
<td></td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>▪ LASTCOMMONQ</td>
<td></td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>▪ LASTQUIESCE (relativeGenerationNumber)</td>
<td></td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>▪ LASTSHUTDOWN</td>
<td></td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>▪ LOGMARK logMarkName (logMarkGeneration)</td>
<td></td>
</tr>
<tr>
<td>(continued)</td>
<td></td>
<td>▪ X’logPoint’</td>
<td></td>
</tr>
<tr>
<td>TOSEQNO</td>
<td>187</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOVOLUME</td>
<td>187</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRANSFORM</td>
<td>171</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPDATE VERSIONS</td>
<td>192</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECOVER UNLOADKEYS</td>
<td>228</td>
<td>ANALYZE</td>
<td>235</td>
</tr>
<tr>
<td>SIMRCVR UNLOADKEYS</td>
<td></td>
<td>CLONE</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP INTABLESPACE</td>
<td>231</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NUMREC</td>
<td>234</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ ABS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ CALC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ EST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ NOEST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>PART</td>
<td>230</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEY</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEDDN</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT</td>
<td>233</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
<td>233</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
<td>230</td>
</tr>
</tbody>
</table>
The section describes the syntax of the OPTIONS command and its options.

**NOTE**

OPTION is also accepted in place of OPTIONS in RECOVER PLUS syntax.

**OPTIONS syntax**

Figure 3 shows the syntax of the OPTIONS command. Figure 4 on page 111, Figure 5 on page 111, and Figure 6 on page 111 show the syntax for the LOGSORT, STOGROUP, and SORT Specifications that are noted in Figure 3. For the conventions that are used in these diagrams, see “Syntax diagrams” on page 24.
Figure 3  OPTIONS syntax diagram (part 2 of 2)
Figure 4  OPTIONS syntax—LOGSORT specification

```
LOGSORT specification

LOGSORT
  page 113

SORTDEV deviceType
  page 114

SORTNUM integer
  page 114

NUMREC
  page 115

CALC

NOEST

EST integer

ABS integer

AVGRECSZ integer

AVGRECSZ integer
```

Figure 5  OPTIONS syntax—STOGROUP specification

```
STOGROUP specification

STOGROUP storageGroupName USEORDER
  page 117

volumeSerialNumber

) ( page 117
```

Figure 6  OPTIONS syntax—SORT specification

```
SORT specification

SORTDIAG
  page 118

NO

YES

SORTDYN
  page 119

YES

NO

RESINV (integerK)
  page 119

SMCORE (integerK, integerK)
  page 119
```
OPTIONS option descriptions

RECOVER PLUS provides options for use with OPTIONS command statements. The options are described in the order in which they are shown in the OPTIONS syntax diagram.

EARLYRECALL / NOEARLYRECALL

The EARLYRECALL and NOEARLYRECALL options determine whether RECOVER PLUS starts early retrieval of archived image copies, log data sets, and change accumulation data sets.

EARLYRECALL causes data sets to be scheduled for recall before other activities begin. A maximum number of tasks is specified during installation by using the RCLTSK option. The data sets are recalled in the order in which RECOVER PLUS accesses them. If RCLTSK is set to zero, the EARLYRECALL option is not allowed. If ANALYZE YES or ANALYZE NO is specified, EARLYRECALL is the default.

NOEARLYRECALL prevents RECOVER PLUS from recalling the data sets until they are accessed or allocated. If ANALYZE ONLY is specified, NOEARLYRECALL is the default. Including EARLYRECALL in an ANALYZE ONLY job causes the data sets that are needed for recovery to be recalled.

---

NOTE

EARLYRECALL/NOEARLYRECALL does not apply to table space or index space data sets. If such data sets are migrated, they are recalled when allocated.

EARLYCAT / NOEARLYCAT

The EARLYCAT and NOEARLYCAT options determine when RECOVER PLUS verifies that cataloged data sets are listed in the operating system catalog. The default value is EARLYCAT.

EARLYCAT enables RECOVER PLUS to verify during the ANALYZE phase that cataloged data sets are listed in the operating system catalog. If the data sets do not exist, RECOVER PLUS determines not to use the data sets during the ANALYZE phase. If other resources are available that allow the recovery to complete normally, they are used.
NOEARLYCAT causes RECOVER PLUS to allow dynamic allocation to determine whether data sets actually exist in the operating system catalog. If a data set does not exist, an error occurs and normal error recovery takes over at that point.

**NOTE**

EARLYCAT/NOEARLYCAT does not apply to table space and index space data sets. These data sets are checked in the ANALYZE phase to determine appropriate actions.

**LOGSORT work data set specification description**

RECOVER PLUS sorts the log and merges the sorted records with copies and change accumulation files. Sorting the log records allows efficient merging of the copies and log records and, if you are rebuilding indexes, permits index keys to be extracted at the same time. By using the LOGSORT specification, you can code values for the SORTDEVT, SORTNUM, and NUMREC options for the sort. You do not need to have log records to use these options (for example, if no updates to the tables and indexes occurred after the most recent copy was made).

If you use the LOGSORT specification values in the OPTIONS command statement, you can also use them in the RECOVER TABLESPACE, RECOVER INDEX, or RECOVER INDEXSPACE command statement; however, this is not necessary.

If you do not use the LOGSORT specification in the OPTIONS command statement and you code multiple RECOVER command statements with the LOGSORT specification on one RECOVER command statement while taking the default on the other RECOVER command statements, the LOGSORT specification values becomes the default for the other command statements. You can code the LOGSORT specification values on more than one of the RECOVER command statements, but it is not necessary.

**NOTE**

All LOGSORT parameters must match if they are used on multiple statements in the same job.

LOGSORT specification values are ignored if the LOGSCAN command is present because LOGSCAN does not invoke a sort.

For more about the LOGSORT option, see “Understanding the LOGSORT strategy” on page 418.
RECOVER PLUS invokes BMCSORT, which allocates the required temporary sort work data sets. You can perform one of the following actions:

■ Specify SORTNUM and SORTDEVT to direct the allocation

■ Specify neither SORTNUM nor SORTDEVT, and let BMCSORT allocate the work data sets according to sort rules

■ Specify neither SORTNUM nor SORTDEVT, and provide L001WKnn DD statements in the JCL

■ Specify only SORTDEVT, and let BMCSORT determine the number of data sets

If you use L001WKnn DD statements in the JCL, any SORTDEVT and SORTNUM specifications that are present are ignored. If you do not use L001WKnn DD statements and do not specify SORTDEVT or SORTNUM, the values that are supplied in the RECOVER PLUS installation options are used.

**SORTDEVT deviceType**

SORTDEVT deviceType specifies the device type for the temporary sort work data sets that BMCSORT uses when it sorts the log. If you do not specify a value, RECOVER PLUS uses the installation option default value.

**SORTNUM integer**

The SORTNUM option affects the allocation of sort work files when BMCSORT is allocating your sort work files dynamically. You can specify an integer value of 1 through 255.

When you specify this option, BMCSORT dynamically allocates the number of sort work files that it needs for each sort task up to the maximum that is illustrated in the following formula:

\[
\text{maximum dynamically allocated sort work files} = n \text{ – preallocated sort work files}
\]

If you specify \textit{integer} from 1 through 32, \(n\) equals 32. If you specify \textit{integer} greater than 32, \(n\) equals \textit{integer}.

---

**NOTE**

Preallocated sort work files include sort work files that are allocated in your JCL.

If you do not specify a value for SORTNUM, RECOVER PLUS uses the installation option default.
NUMREC

NUMREC specifies the size of the sort file for the log sort routine. The size is defined as an estimate or the exact number of log records to process and the estimated average length of each log record. For information about performance implications, see Chapter 6, “RECOVER PLUS concepts.”

NUMREC CALC

NUMREC CALC is the default value. Specifying this option allows RECOVER PLUS to estimate the number of log records to be processed by the log sort routine. It estimates that the sort receives one log record per control interval (CI) processed.

NUMREC NOEST

Specifying NUMREC NOEST prevents RECOVER PLUS from passing a file size to the log sort routine.

NUMREC EST integer

Use NUMREC EST integer to specify an estimated number of log records for the log sort routine. integer must be a positive integer.

AVGRECSZ integer

Use AVGRECSZ integer to specify the average length of the log records to sort. If you do not specify AVGRECSZ, RECOVER PLUS determines its value.

NUMREC ABS integer

Use NUMREC ABS integer to specify the exact number of log records for the log sort routine. integer must be a positive integer.

WARNING

If the integer parameter is a value that does not represent the true number of log records that are passed to the log sort routine, the log sort routine abends. Use extreme caution when you specify NUMREC ABS.

AVGRECSZ integer

Use AVGRECSZ integer to specify the average length of the log records to sort. If you do not specify AVGRECSZ, RECOVER PLUS determines its value.
OPTIONS option descriptions

NOSYSLGRNG

NOSYSLGRNG causes the SYSIBM.SYSLGRNX table to be bypassed when determining what logs are used during recovery.

NOSYSLGRNG is used automatically when the object to be recovered has been dropped and is being recovered with the DROPRECOVERY option (see page 170). If you have a damaged SYSIBM.SYSLGRNX table, you can use the NOSYSLGRNG option to avoid accessing those tables. This option causes all log records to be scanned without regard to log ranges, using SYSCOPY entries and end points to determine what log is needed.

Indexes that are defined as COPY NO do not register SYSIBM.SYSLGRNX entries. Recoveries of COPY NO indexes are effectively NOSYSLGRNG recoveries.

USEACCUM

The USEACCUM option indicates whether to access change accumulation files. How USEACCUM works depends on the availability of a valid password for the BMC R+/CHANGE ACCUM product or BMC Recovery Management for DB2 solution as shown in the following table:

<table>
<thead>
<tr>
<th>USEACCUM value</th>
<th>With a valid R+/CHANGE ACCUM or Recovery Management password</th>
<th>Without a valid password</th>
</tr>
</thead>
<tbody>
<tr>
<td>not specified</td>
<td>RECOVER PLUS looks for and uses available change accumulation files.</td>
<td>RECOVER PLUS does not look for or use change accumulation files.</td>
</tr>
<tr>
<td>USEACCUM YES</td>
<td>RECOVER PLUS looks for and uses available change accumulation files.</td>
<td>RECOVER PLUS looks for and uses available change accumulation files.</td>
</tr>
<tr>
<td>USE ACCUM NO</td>
<td>RECOVER PLUS does not look for or use change accumulation files.</td>
<td>RECOVER PLUS does not look for or use change accumulation files.</td>
</tr>
</tbody>
</table>

USEACCUM YES

If the R+/CHANGE ACCUM product is installed, USEACCUM YES is the default.

USEACCUM NO

If R+/CHANGE ACCUM is not installed, USEACCUM NO is the default.
CHECKINT *(integer)*

This option specifies the time in minutes between checkpoints. Use this option to balance frequency of restart points with the overhead incurred when taking restart points. When you specify CHECKINT on the OPTIONS command statement, the value specified overrides the value of the CHECKINT installation option (which defaults to 0).

If the installation option CHECKPT is set to CHECKPT=PHASE, a restart point is taken only if more minutes than those specified by CHECKINT have elapsed since the last restart point was taken. Use CHECKINT 0 (the default value) on the OPTIONS command statement and the CHECKPT installation option set to CHECKPT=PHASE to force a restart checkpoint after every phase.

STOGROUP specification description

Use the STOGROUP specification to control the order in which volumes in a STOGROUP are allocated during the execution of RECOVER PLUS. Because a STOGROUP definition in the DB2 catalog has no inherent ordering of the volumes, a STOGROUP specification with the OPTIONS command statement becomes necessary when you need a specific order.

Allocations for table spaces and index spaces are attempted on volumes in the order specified in the USEORDER clause. Volumes in the DB2 storage group that are not included in the USEORDER clause are used after those explicitly requested and are used in the order that they are retrieved from SYSIBM.SYSVOLUMES.

You can include more than one STOGROUP specification in one OPTIONS command statement.

If you do not use the STOGROUP specification, the volumes are allocated in the order that they are retrieved from SYSIBM.SYSVOLUMES. For more information, see “Storage group-defined data sets” on page 92.

**STOGROUP storageGroupName**

STOGROUP *storageGroupName* specifies the name of a DB2 storage group. *storageGroupName* must be the name of a storage group that is found in SYSIBM.SYSSTOGROUP. If the storage group is not found in SYSIBM.SYSSTOGROUP, a warning is issued during analysis. If a valid storage group is included in the STOGROUP specification but is not referenced by any of the spaces being recovered, a warning is issued.
**USEORDER volumeSerialNumber**

USEORDER *volumeSerialNumber* specifies a list of one or more volume serial numbers, separated by commas. Left and right parentheses are required around the list. Each *volumeSerialNumber* in the list must be found in the SYSIBM.SYSVOLUMES associated with *storageGroupName*. If a volume serial number that is specified is not found in SYSIBM.SYSVOLUMES, a fatal error occurs during analysis.

In the following OPTIONS command statement, all table spaces implicated in the recovery that use STOGROUP PRODSG1 will try to find space on the volumes in the order PROD01, PROD02, PROD03, and PROD04. Table spaces that use STOGROUP PRODSG2 will use volumes PROD02, PROD03, PROD04, and PROD01 (in that order).

**NOTE**

RECOVER PLUS uses BMCSORT as the sort engine. **BMC recommends that you not change any SORT specification option defaults unless necessary.**

**SORT specification description**

Use the SORT specification to have greater control over the sort routine. You can specify values for the parameters that are associated with the SORT specification only on the OPTIONS command statement in RECOVER PLUS. Your specifications affect all sorts invoked by the recovery. (For more information, see “Managing sort performance” on page 433.)

**SORTDIAG**

Use the SORTDIAG option to provide sort diagnostic messages.

**SORTDIAG NO**

SORTDIAG NO is the default and turns off any provision for diagnostic messages.

**SORTDIAG YES**

SORTDIAG YES turns on diagnostic messages. The messages are written to SYSOU*nnn*. These messages include storage usage statistics and EXCP counts. RECOVER PLUS dynamically allocates a SYSOU*nnn* DD statement automatically if one is not provided.
SORTDYN

Use the SORTDYN option to control and override installation defaults for sort work dynamic allocation.

SORTDYN YES

SORTDYN YES specifies that dynamic allocation can occur and should use the RECOVER PLUS specification for SORTDEVT and SORTNUM.

SORTDYN NO

SORTDYN NO specifies that BMCSORT defaults for sort work dynamic allocation should be used.

RESINV (integerK)

This option overrides the RESINV installation option (see “RESINV=0K” on page 546) and specifies the amount of memory that BMCSORT reserves below the 16-MB line to allow for IDCAMS processing. The installation default value of RESINV=0K allows BMCSORT defaults to be used.

**NOTE**

BMC recommends that you use the default value so that BMCSORT uses its algorithms effectively.

SMCORE (integerK, integerK)

This option overrides the SMCORE installation option (see page 545) and specifies the maximum amount of memory that BMCSORT uses. The first value specifies the total amount of memory used both above and below the 16-MB line. The second value specifies the amount of memory used below the 16-MB line. The default values of SMCORE=(0K,0K) allow BMCSORT defaults to be used.

**NOTE**

BMC recommends that you use the SMCORE=(0K,0K) defaults so that BMCSORT uses its algorithms effectively.

For more information, see “Managing sort performance” on page 433.
RESOURCESSELECTION

Use the RESOURCESSELECTION option to indicate a preferred order in the selection of image copies, logs, and change accumulation files.

RESOURCESSELECTIONLOGS

You can specify RESOURCESSELECTION as ACT1, ACT2, ARC1, and ARC2 in any order. ACT1 and ACT2 indicate the primary and dual (secondary) active logs. ARC1 and ARC2 indicate the primary and dual (secondary) archive logs. The default order is ACT1, ACT2, ARC1, ARC2. You can omit references to log copies that you do not want considered. However, if you omit references to active logs, they may be used anyway if the required log ranges are not yet archived.

RESOURCESSELECTIONACCUMS

You can specify RESOURCESSELECTION as LP, LB, RP, and RB in any order. LP and LB indicate the primary and secondary local change accumulation files. RP and RB indicate the primary and secondary remote change accumulation files. The default order is LP, LB when operating as a local site and RP, RB when operating in a recovery site. You can omit references to copies of the resource that you do not want considered.

RESOURCESSELECTIONCOPIES

You can specify RESOURCESSELECTION as LP, LB, RP, RB, and FC in any order. LP and LB indicate the primary and secondary local image copies. RP and RB indicate the primary and secondary remote image copies. FC indicates an IBM FlashCopy. The default order is FC, LP, LB when operating as a local site, and RP, RB, FC when operating in a recovery site. You can omit references to copies of the resource that you do not want considered.

MAXDRIVESinteger

The MAXDRIVES option controls the number of tape drives that are used during recovery. If a value for MAXDRIVES is not coded, the limit specified by the MAXDRIVE installation option is used.

If more drives are required for the MERGE processing than the MAXDRIVES value, RECOVERPLUS does not start the process. If MAXDRIVES is less than MAXLOGS, no more than the number of MAXDRIVES tape log data sets are opened at one time.
MAXLOGS integer

The MAXLOGS option is part of the concurrent log file reading feature. Use this option to perform the following tasks:

- control the amount of memory that is used during the recovery
- reduce the contention that is caused by reading many log files in parallel

MAXLOGS must be greater than 0.

When MAXLOGS is specified, RECOVER PLUS will not allocate and read more log files concurrently than indicated by this option. If MAXLOGS is not coded on the OPTIONS command statement, the default is the value of the MAXLOGS installation option.

In addition to the global control that is provided by MAXDRIVES, use MAXLOGS to further control the number of tape drives that are used for the log files. If the value of MAXLOGS is less than MAXDRIVES, the number of tape drives that are used for log files is limited to the value of MAXLOGS.

OUTCOPY

OUTCOPY specifies how image copies are created for partitioned objects. The presence of OUTCOPY on the OPTIONS command statement overrides, for this recovery, the setting of OUTCOPY in the installation options.

Output copies are produced in the SYSTEMPAGES YES format when possible.

OUTCOPY ASCODED

OUTCOPY ASCODED designates that image copies are created as specified for the RECOVER TABLESPACE, RECOVER INDEX, or RECOVER INDEXSPACE command statement. When DSNUM ALL is specified, an image copy is created for the object as a whole. If the recover specification is for an individual data set (DSNUM n), a copy is made of that specific data set.

OUTCOPY BYPART

OUTCOPY BYPART designates that each partition of a partitioned object is copied to a separate image copy.
The AUX option specifies if auxiliary objects will be included with the recovery of the base table spaces.

The AUX option is available on the OPTIONS command or in the table space specification for any of the following RECOVER PLUS commands:

- RECOVER TABLESPACE
- RECOVER INDEX(ALL) TABLESPACE
- REBUILD INDEX(ALL) TABLESPACE
- LOGSCAN TABLESPACE
- LOGSCAN INDEX(ALL) TABLESPACE

If you do not specify AUX on one of these commands, RECOVER PLUS uses the value of the AUX installation option (page 555), which has a default value of NO.

**NOTE**
RECOVER PLUS ignores the AUX option when you specify OBJECTSET.

Also, the AUX option has no effect for the ACCUM TABLESPACE, RECOVER UNLOADKEYS, and RECOVER BUILDINDEX commands.

When you specify AUX YES, AUX XML, AUX LOB, or AUX HISTORY, you cannot specify the following options:

- TOCOPY `dataSetName`
- OBIDXLAT without defaults
- OUTCOPYDSN
- RECOVERYDSN

Valid values are NO, YES, XML, LOB, and HISTORY.

**AUX NO**
RECOVER PLUS does not include any auxiliary objects in the recovery.

**AUX YES**
RECOVER PLUS includes both LOB and XML objects in the recovery, as well as history spaces when a table space with system-maintained temporal tables is specified in SYSIN.
AUX XML

RECOVER PLUS includes all XML data objects along with the XML base table space. If the table space specification is found on a RECOVER TABLESPACE or SIMULATE TABLESPACE statement, all implicitly created XML table spaces will be processed along with the base XML table space. If the table space specification is found on a RECOVER INDEX(ALL), REBUILD INDEX(ALL), or SIMULATE INDEX(ALL) statement, all implicitly created XML and node ID indexes on the XML base table space as well as all explicitly created non-LOB indexes will be processed.

AUX LOB

RECOVER PLUS includes all LOB data objects along with the LOB base table space. If the table space specification is found on a RECOVER TABLESPACE or SIMULATE TABLESPACE statement, all table spaces created with the LOB attribute will be processed along with the base LOB table space. If the table space specification is found on a RECOVER INDEX(ALL), REBUILD INDEX(ALL) or SIMULATE INDEX(ALL) statement, all indexes on the LOB base table space and all non-LOB explicitly created indexes will be processed.

AUX HISTORY

When you specify AUX HISTORY, RECOVER PLUS includes history spaces when a table space with system-maintained temporal tables is specified in SYSIN.

Similar to DB2 RECOVER Version 10 VERIFYSET NO behavior, RECOVER PLUS does not check for the following:

- that the system-maintained temporal space and the history space are recovered as a set
- that the recovery of the system-maintained temporal space and the history space is to the same log point
- that all DSNUMs are recovered in the same SYSIN

USEHDROBIDS

Use the USEHDROBIDS option to indicate if the OBIDs in the header are valid or not.

When you specify OBIDXLAT (page 193) with INCOPY FULL SNAPSHOT TOCOPY syntax and do not specify OBIDs, RECOVER PLUS looks at the OBIDs in the header page to determine if the OBIDs for the source and target are the same. If the OBIDs are the same, RECOVER PLUS does not use a MERGE phase to update the OBIDs. Skipping the MERGE phase makes migration faster because it eliminates a read and write of every page of the space. (RECOVER PLUS may still do an “optimized merge” phase to just update the header page.)
Sometimes the OBIDs in the header page are not correct. An example of when this situation might occur is when the space is copied from another object and the OBIDs in the header page are not translated correctly. If the OBIDs in the header page are not correct, RECOVER PLUS might skip the MERGE phase when it should not, or the MERGE phase might fail because RECOVER PLUS detects that the OBIDs in the header page do not match the OBIDs in the rest of the data. Under circumstances like these, you can use USEHDROBIDS to tell RECOVER PLUS if the OBIDs in the header are valid or not.

If you do not code USEHDROBIDS on the OPTIONS command statement, the default is the value of the USEHDROBIDS installation option (page 557), which defaults to YES.

**USEHDROBIDS YES**

When you specify USEHDROBIDS YES, RECOVER PLUS uses the OBIDs in the header page and skips the MERGE phase if possible. If RECOVER PLUS does perform the MERGE phase and detects that the OBIDs are not correct, RECOVER PLUS ends with an error message.

**USEHDROBIDS NO**

When you specify USEHDROBIDS NO, RECOVER PLUS always does a MERGE phase and ignores the OBIDs in the header page.

**BACKOUT**

The BACKOUT option invokes the backout strategy for point-in-time recovery by using log points (TORBA, TOLOGPOINT). This strategy assumes that spaces are undamaged and that you require a reset to a specific point in time. The spaces and the log records between the point in time and the current point are used to back out to the required state.

**NOTE**

The space must not be in RECP, RECP*, RBDP, RBDP*, PSRCP, PSRBD, GRECP, WEPR, or STOPE status or have an LPL range.

BACKOUT also cannot be used for LOB spaces or for spaces having a logging attribute of NOT LOGGED.

If you do not specify BACKOUT on the OPTIONS command, RECOVER PLUS uses the value of the BACKOUT installation option (page 556), which has a default value of AUTO. If you are not using a Recovery Management password, point-in-time recoveries use a standard forward recovery.
The BACKOUT option causes all point-in-time recoveries specified in the SYSIN to use the backout strategy. If the BACKOUT option is specified on the OPTIONS command statement and all of the RECOVER command statements in the SYSIN are for a recovery to the current point in time, the BACKOUT option is ignored.

If you specify the BACKOUT option on the OPTIONS command statement and the SYSIN contains at least one object being recovered to a specific point in time, all objects being recovered in the SYSIN must be recovered to a point in time. If none of the objects are being recovered to a point in time, BACKOUT on the OPTIONS command statement is ignored. For a detailed discussion of the backout recovery strategy, see “Strategies for point-in-time recovery” on page 426.

**NOTE**

If a table space is recovered to a specific point in time by using BACKOUT and an associated index is recovered without specifying a TOLOGPOINT, TORBA, or TOCOPY keyword in the same SYSIN, the index is recovered with BACKOUT to the same point in time as the table space.

**BACKOUT AUTO**

BACKOUT AUTO supports the Recovery Management for DB2 backout to forward recovery feature.

The BACKOUT AUTO option invokes the recovery strategy in which all of the PIT recovery requests are first executed using the BACKOUT option. If any BACKOUT request fails, forward recovery is performed for the objects that were not backed out. This automation produces the fastest possible recovery with minimal intervention and is only available using the Recovery Management for DB2 solution.

**NOTE**

- The BACKOUT AUTO option requires a Recovery Management solution password.
- Generate BACKOUT AUTO syntax by using the RECOVERY MANAGER component of the Recovery Management solution.

For more information BACKOUT AUTO recovery, see the *Recovery Management for DB2 User Guide*.

**BACKOUT NO**

BACKOUT NO specifies that you want to perform standard forward point-in-time recoveries.
BACKOUT YES

BACKOUT YES invokes the backout strategy for point-in-time recovery by using log points (TORBA, TOLOGPOINT). This strategy assumes that spaces are undamaged and that you require a reset to a specific point in time. The spaces are used with the log records between the point in time and the current point to back out to the required state. Using BACKOUT may enhance the performance of a point-in-time recovery significantly. However, if the backout recovery fails, RECOVER PLUS stops with an error. (As opposed to BACKOUT AUTO, which will move to forward recovery if the backout recovery fails.) For more information, see “Using BACKOUT” on page 426.

The following restrictions apply:

- The space must be current as of the last logged activity and not damaged in any way. Multiple-data set, nonpartitioned spaces must have all data sets scanned (DSNUM ALL).

- No LOAD or REORG events can exist between the log point specified and the current log points. For indexes, no REBUILD INDEX events can exist in this range.

- No prior point-in-time recovery with a START_RBA greater than the log point requested and a PIT_RBA less than the log point requested can exist.

- BACKOUT may not be requested with LOGONLY or LOGAPPLY ONLY.

**NOTE**

BACKOUT uses only logs and spaces but requires that the spaces be current. The LOGONLY and LOGAPPLY ONLY options imply scanning log records going forward by using a space restored to a previous state.

- Change accumulation files are not allowed with BACKOUT because they are not properly ordered. Output accumulation files are also not supported because they are defined from the point of the last image copy to the current point.

INDEXLOG

The INDEXLOG option setting determines the behavior of the RECOVER INDEX command. If it is not specified on the OPTIONS command statement, the setting of INDEXLOG in the installation options is used.
You might consider using INDEXLOG YES, even though you may have to change some RECOVER INDEX commands to REBUILD INDEX commands. However, if you have no immediate plans to start making index image copies, you may prefer INDEXLOG NO so that you will not have to change syntax.

**NOTE**
The RECOVER INDEXSPACE command always invokes a recovery that is based on copies and logs, regardless of the INDEXLOG setting. REBUILD INDEX always rebuilds indexes, regardless of the INDEXLOG setting.

**INDEXLOG YES**

INDEXLOG YES causes the RECOVER INDEX command to use copies, log, or both for index recovery.

**INDEXLOG NO**

INDEXLOG NO causes the RECOVER INDEX command to rebuild the index by extracting the keys from the table space. When you specify INDEXLOG NO, the RECOVER INDEX and the REBUILD INDEX commands are synonyms.

**NOTE**

In this book, references to REBUILD INDEX may be replaced with RECOVER INDEX if INDEXLOG NO is specified.

**INDEXLOG AUTO**

INDEXLOG AUTO causes the RECOVER INDEX command to recover the index from image copies and log if possible. However, if RECOVER PLUS determines that it cannot recover the index (for example, because of missing image copy or a point-in-time recovery), the RECOVER INDEX request automatically converts to a REBUILD INDEX request. INDEXLOG AUTO supports BACKOUT recoveries.

Conversion from RECOVER INDEX to REBUILD INDEX is not supported for the following options:

- **OUTCOPY**: The OUTCOPY option is not supported for REBUILD INDEX so this request is not eligible for conversion.

- **OBIDXLAT**: The OBIDXLAT option is not supported for REBUILD INDEX so this request is not eligible for conversion.

- **TOLOGPOINT**: If the TOLOGPOINT option is included in the RECOVER INDEX request, the request is only eligible for conversion to REBUILD INDEX if the associated table space is also recovered to the same log point in the same job step.
- TOCOPY: If the TOCOPY option was included in the RECOVER INDEX request, the request is only eligible for conversion to REBUILD INDEX if the associated table space is also recovered in the same job step.

- DSNUM: If the RECOVER INDEX command was for a specific data set of a nonpartitioned index, the request is not eligible for conversion to REBUILD INDEX.

**IXRECP**

When you are performing a point-in-time recovery, the IXRECP option setting determines whether indexes that are not rebuilt or recovered in the same run as their associated table spaces are set to RECP or RBDP status.

Use the IXRECP option when a point-in-time recovery of a table space is performed. In that type of recovery, it is important that any indexes on the space are rebuilt or recovered so that index data is synchronized with the data to which it refers. You can use IXRECP to have the pending status set for the indexes that are associated with the recovered table spaces, which forces index recoveries before the data can be accessed.

If IXRECP is not present in the OPTIONS command statement, the setting of IXRECP is determined by the IXRECP installation option.

When you are running Recovery Management and using the TRANSFORM option, you need the IXRECP option because any table space that is transformed must either have the index transformed or rebuilt. The RBDP status will make it obvious if a target index does not reflect the changes made to the table space.

**IXRECP YES**

IXRECP YES tells RECOVER PLUS to issue a warning message for each index that has not been recovered with the table space and to put each such index in RECP or RBDP status. A later index recovery or rebuild removes the RECP or RBDP status. If indexes are rebuilt or recovered in the same run as the table space, they are not placed in RECP or RBDP status.

**IXRECP NO**

IXRECP NO, the default, causes RECOVER PLUS not to set the RECP or RBDP status for indexes that are associated with a table space that is recovered to a point in time. No message is issued regarding the indexes.
TRTCH

The TRTCH option allows the use of Improved Data Recording Capability (IDRC) tape compression when creating output change accumulation files.

TRTCH NONE

This value is the default and specifies the use of the system default for the allocation of output change accumulation files. TRTCH NONE is equivalent to not specifying TRTCH.

TRTCH COMP

TRTCH COMP specifies that IDRC compression is to be used for output change accumulation files.

TRTCH NOCOMP

TRTCH NOCOMP specifies no IDRC compression for output change accumulation files (even if the system default is TRTCH COMP).

XBMID ssid or xbmGroup

The XBMID option specifies the EXTENDED BUFFER MANAGER (XBM) subsystem ID (ssid) or XBM group name (xbmGroup) to use when you restore Instant Snapshot copies or when you use the zIIP redirection capability.

ssid is the unique identifier that you specified when you installed XBM. If you are using XBM in a DB2 data sharing environment, you can use the xbmGroup name in place of ssid. The xbmGroup value is the name of the XBM coupling facility group defined to the XBM subsystem.

For the zIIP redirection capability, if you specify an XBM subsystem and ZIIP ENABLED (page 130 and page 547) is in effect, RECOVER PLUS attempts to use that subsystem to enable zIIP processing. If that subsystem is not available or if it is not at the correct maintenance level, zIIP processing is not enabled.

If you do not specify an XBM subsystem either with XBMID on the OPTIONS command or with the XBMID installation option, RECOVER PLUS searches for an XBM subsystem at the appropriate maintenance level to enable zIIP processing.

For recovery using Instant Snapshots, RECOVER PLUS does not discover the XBM subsystem. Use the XBMID option in either the installation options module or on the OPTIONS statement to specify it. Instant Snapshots are made by COPY PLUS with XBM and are registered in BMCXCOPY. For more information, see the COPY PLUS for DB2 Reference Manual.
You use XBMID on the OPTIONS command to override the XBMID installation option (see “XBMID=ssid or xbmGroup” on page 546).

For specific information about valid characters for XBMID and the pattern matching capabilities of XBM, Instant Snapshots, and zIIP redirection, see the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide.

ZIIP

The ZIIP option tells RECOVER PLUS whether to attempt to use IBM® System z® Integrated Information Processors (zIIPs). RECOVER PLUS can use enclave service request blocks (SRBs) to enable zIIP processing automatically while running jobs. Using zIIP processing can reduce the overall CPU time for RECOVER PLUS jobs.

You can specify the default for the ZIIP command option in your installation options module by using the ZIIP installation option (page 549). RECOVER PLUS was shipped with a default value of ENABLED for this option. The ZIIP option on the OPTIONS command overrides the default that is in the installation options module.

ZIIP ENABLED

ZIIP ENABLED tells RECOVER PLUS to attempt to offload eligible processing to an available zIIP. If the zIIP is busy or not available, normal processing continues on a general-purpose processor.

To enable and use zIIP processing with RECOVER PLUS, you must

- have an installed authorized version of XBM or SUF
- start and maintain an XBM subsystem in your environment
- have a zIIP available in your environment

You can specify a particular XBM subsystem to use by specifying a value for the XBMID installation option (page 546) or XBMID option on the OPTIONS command (page 129).

XBM and SUF are licensed, installed, and maintained separately from RECOVER PLUS. You can use either XBM or SUF, depending on the license that you have obtained:

- A license for the full version of the XBM product authorizes you to use all features of XBM.
- A license for SUF authorizes you to use only the snapshot and zIIP-processing features of XBM.

For more information about XBM and SUF, see the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide.
ZIIP DISABLED

ZIIP DISABLED tells RECOVER PLUS to not attempt to use zIIP processing.

**DATAMVR programName**

Use the DATAMVR option to override the DATAMVR installation option value. The DATAMVR installation option provides XBM with the name of the program to use to copy a data set if a data set snap fails. To use DFDSS as the data mover, specify DATAMVR=ADRDSSU.

**KEYSORT**

Use the KEYSORT option to override the SORTDEVT and SORTNUM values in the installation module for all REBUILD INDEX or RECOVER UNLOADKEYS index sorts within a job step. The values that are specified in the OPTIONS command with KEYSORT can be overridden by SORTDEVT and SORTNUM on the REBUILD INDEX or RECOVER UNLOADKEYS command statements.

When performing a REBUILD INDEX or a RECOVER UNLOADKEYS job, RECOVER PLUS invokes BMCSORT, which allocates the required temporary sort work data sets. You can specify one of the following courses of action:

- specify SORTNUM and SORTDEVT to direct the allocation
- specify neither SORTNUM nor SORTDEVT and let BMCSORT allocate the work data sets according to sort rules
- specify neither SORTNUM nor SORTDEVT and specify SxxxWKnn DD statements in the JCL
- specify only SORTDEVT and let BMCSORT determine the number of data sets

If you use SxxxWKnn DD statements in the JCL, any SORTDEVT and SORTNUM specifications that are present are ignored. If you do not use SxxxWKnn DD statements and do not specify SORTDEVT or SORTNUM, the values that are supplied in the RECOVER PLUS installation options module are used.

**SORTDEVT deviceType**

SORTDEVT `deviceType` specifies the device type for the temporary sort work data sets that BMCSORT uses when it sorts the index keys.

If you specify SORTDYN NO (page 119), BMCSORT defaults are used even if you specify a value for SORTDEVT. Otherwise, RECOVER PLUS uses its internal default value, SYSDA, if you do not specify SORTDEVT.
**SORTNUM integer**

The SORTNUM option affects the allocation of sort work files when BMCSORT is allocating your sort work files dynamically. You can specify an integer value of 1 through 255.

When you specify this option, BMCSORT dynamically allocates the number of sort work files that it needs for each sort task up to the maximum that is illustrated in the following formula:

\[
\text{maximum dynamically allocated sort work files} = n - \text{preallocated sort work files}
\]

If you specify \textit{integer} from 1 through 32, \( n \) equals 32.

If you specify \textit{integer} greater than 32, \( n \) equals integer.

---

**NOTE**

Preallocated sort work files include sort work files that are allocated in your JCL.

If you do not specify a value for SORTNUM, RECOVER PLUS uses the installation option default.

**ANALYZE**

The ANALYZE option prints a recovery plan before executing that plan. The following information about objects is included in the plan:

- names of any image copy data sets on which activities are or will be based
- names of any log data sets on which activities are or will be based
- names of any change accumulation files on which activities are or will be based
- log ranges, if any, on which activities are or will be based
- number of log pages to be read
- record sizes for index sort work data sets
- phases to occur during execution
- steps to occur within each phase

You can use this information to allocate space more accurately for log sort work data sets, index sort work data sets, and index key work data sets, and limit abends that are caused by inadequate data set allocations. You can also use these statistics with historical information about the times that are required to perform the various operations to estimate recovery time.
ANALYZE YES

ANALYZE YES is the default and provides information about the objects in the recovery plan.

ANALYZE NO

If you specify ANALYZE NO, RECOVER PLUS does not provide any recovery plan information, but does provide object summaries.

ANALYZE ONLY

If you specify ANALYZE ONLY, RECOVER PLUS provides the same information as it provides with ANALYZE YES and then stops the job when the ANALYZE phase is complete. You can use the data from the ANALYZE phase to determine the resources required and what will happen during your RECOVER PLUS run. You cannot restart a RECOVER PLUS job that specifies ANALYZE ONLY, but you can start a new recovery job.

Use the ANALYZE ONLY option to determine which phases will occur with a specific recovery request, or which copy data sets and log data sets will be used in the recovery.

If you provide a SYSPICK DD statement, RECOVER PLUS generates a list of all of the input tape and cartridge volumes that are allocated during recovery. For more information about SYSPICK, see “RECOVER PLUS data sets and RECOVER PLUS DD statements” on page 276.

If you specify OPTIONS EARLYRECALL with ANALYZE ONLY, the recall of the data sets needed for the recovery will be initiated. You must code OPTIONS EARLYRECALL explicitly to initiate the recalls because OPTIONS NOEARLYRECALL is the default for ANALYZE ONLY executions.

Use ANALYZE ONLY on the OPTIONS command for the recovery estimation feature available in the Recovery Management for DB2 solution. The estimation information generated when you use this option in Recovery Management is stored in DB2 recovery history tables. For more information, see the Recovery Management for DB2 User Guide.

NOTE

The value that is specified for ANALYZE is effective for all RECOVER and REBUILD commands that follow. You can code the ANALYZE option on individual commands, but the value that you code must match the value that is coded on the OPTIONS command. The LOGSCAN command ignores the ANALYZE option.
The SIMULATE YES option specifies that you want to run in simulation mode, which provides a way for you to validate that you can recover your application data. You can verify that needed recovery resources are valid and available and that log apply can be done.

**NOTE**
Simulation is a feature of the Recovery Management for DB2 solution and requires a valid Recovery Management solution password.

In simulation mode, RECOVER PLUS proceeds with all recovery activities that are not destructive. Destructive actions that could affect real database objects or recovery environment are simulated or suppressed.

There are several limitations and requirements you should take in account when you use the SIMULATE YES option:

- The SIMULATE YES option cannot be specified with the ANALYZE ONLY, or ANALYZE SCANONLY, or BACKOUT options.

- The LOGSCAN command cannot be used if you specify the SIMULATE YES option.

- Point-in-time recovery is not allowed if the SIMULATE YES option is specified.

- BACKOUT, LOGAPPLY, or LOGONLY specifications in recovery command statements are not allowed if the SIMULATE YES option is specified.

- The requested recovery scenario can require unloading keys from the real table space (partition) if REBUILD or UNLOADKEYS is requested for the table space (partition) not recovered in the same step. Such actions will be suppressed if the SIMULATE YES option is specified.

- INLINE copies or SNAP copies will not be used as recovery resource if the SIMULATE YES option is specified.

- Objects may be in any initial status and there is no impact on the status or contents of the real database objects if the SIMULATE YES option is specified.

- No registration into DB2 or BMC tables occurs and no restarts are possible if the SIMULATE YES option is specified.

- Some options related to the suppressed or simulated actions will have no impact if the SIMULATE YES option is specified.
Any output activity for the spaces and output copies (including data set allocation, creation, and extension) is only simulated if the SIMULATE YES option is specified. No real I/O or requests to the system services take place.

Use SIMRCVR and SIMRBLD instead of regular RECOVER and REBUILD if the SIMULATE YES option is specified.

For more information about recovery simulation, see *Recovery Management for DB2 User Guide*. You can use RECOVERY MANAGER (RMGR) for DB2 and its online interface to generate JCL for the simulation mode for both application and system recovery.

**SIMULATE YES LOGPOINT X’logPoint’**

Use the SIMULATE YES LOGPOINT X’logPoint’ option to specify the "pseudo current" point for the simulated recovery. A request to simulate recovery to the current point is done if LOGPOINT X’logPoint’ is omitted.

**WTOR**

If one or more spaces in the RECOVER command statement remain in STOPP status, WTOR provides the flexibility to issue a write to the operator requesting a reply (WTOR YES) or assume an operator reply of CANCEL (WTOR NO) and terminate the job.

**WTOR YES**

RECOVER PLUS issues a WTOR.

**WTOR NO**

RECOVER PLUS assumes an operator reply of CANCEL and terminates the job.

**ON ERROR ANY CONTINUE integer or ON ERROR CONTINUE integer**

The ON ERROR ANY CONTINUE integer option enables you to specify how RECOVER PLUS is to proceed when errors are encountered.

ANY is optional and is the default value. ANY specifies that the requested action is to be taken for any recognized severe error. This value is provided to allow selection by type of error in the future.

The integer variable can be any integer value between 0 and 2,147,483,646.
ON ERROR CONTINUE integer allows integer + 1 errors before RECOVER PLUS terminates. If integer is 0, RECOVER PLUS stops processing immediately when the first recognized severe error occurs.

If ON ERROR CONTINUE 0 is specified, the subtask to preallocate VSAM data sets is disabled. This could increase the execution time by several seconds for each object recovered. If BACKOUT AUTO is specified through use of the Recovery Management solution, RECOVER PLUS does not limit the number of errors allowed. For information about the BACKOUT AUTO option, see “Using BACKOUT AUTO recovery” on page 61 or the Recovery Management for DB2 User Guide.

**MAXKSORT integer**

The MAXKSORT option specifies the maximum number of index key sorts and index rebuilds that can run in parallel subtasks.

---

**NOTE**

When you rebuild the indexes of a multi-data-set, nonpartitioned table space, the UNLOADs run serially in the main task, but the REBUILDs are multitasked.

---

Valid values are 1 to 999. If you do not specify a value for MAXKSORT, RECOVER PLUS uses the value of the MAXKSORT installation option, which defaults to the following formula:

\[
\text{minimum}(2 \times \text{the number of CPUs}), 12
\]

---

**NOTE**

Each sort requires about 256 KB of memory below the line. Values for MAXKSORT greater than 12 are not recommended.

---

For a description of the MAXKSORT installation option, see “MAXKSORT” on page 544.

---

**NOTE**

When the value of MAXKSORT is greater than 1, RECOVER PLUS ignores the WORKDDN option on the REBUILD INDEX command and issues a warning message. To use the WORKDDN option, specify a value of 1 for MAXKSORT. For a description of WORKDDN, see page 221.

---

Using the MAXKSORT option can improve recovery performance.
The MAXKSORT value determines the level of concurrency that can be achieved for index key sorts and index rebuilds. If this value is too small, the level of concurrency could be unnecessarily limited and the size of each sort will be larger. If this value is too large, the recovery job could overuse system resources and degrade recovery performance and overall system performance.

The concurrency that is reached by using MAXKSORT is limited by the following items:

- the amount of memory that is available below the 16-MB line for BMCSORT processing

  In most environments, available memory below the 16-MB line creates a practical limit of 15 to 20 sorts.

- the value assigned to the KSORTSHARE option (page 138 and page 554)

  KSORTSHARE specifies if key sorts are shared among RECOVER PLUS table space recoveries (MERGE phases) running in parallel.

If the key lengths of the indexes vary widely in size, MAXKSORT increases efficiency because it allocates only the amount of memory that is needed for each key.

If the index rebuild includes both partitioned and nonpartitioned indexes, MAXKSORT, if set to a value of 3 or greater, could allow the sorting of the partitioned indexes separately from the nonpartitioned indexes and might improve efficiency.

For each table space, index keys for all indexes being rebuilt are distributed over the number of sorts that you specify for this option and these sorts can run in parallel. If RECOVER PLUS is recovering a partitioned table space and is rebuilding the partitioning index, the rebuild of each partition may be performed at the completion of the MERGE for each partition of the table space. The rebuild can run concurrently with the MERGE for the next partition if the MAXKSORT number is not exceeded. Running concurrent index key sorts and index rebuilds can increase the speed of the recovery.

Restart can cause keys that have already been extracted and sorted to be extracted and sorted again, but the restart process is relatively straightforward.

The following files are dynamically allocated if you do not code them in JCL:

- SYSOU*nnn: sort message files

  *nnn* is a number between 1 and the value specified for MAXKSORT.
- **SxxxWKnn**: key sort work files

  `xxx` is a number between 1 and the value specified for MAXKSORT. `nn` is a number between 1 and the value specified for the SORTNUM installation option or the OPTIONS SORTNUM parameter.

  When you use dynamic allocation for these files, BMCSORT determines the optimal number of files to use.

  For more information about setting MAXKSORT, see “Setting the MAXKSORT option” on page 439.

### MAXLSORT integer

The MAXLSORT option specifies how many log sorts RECOVER PLUS can run in parallel and also determines the number of MERGE/SNAP/RESTORE phases that can run in parallel regardless of log requirements. These phases run in subtasks and MAXLSORT sets the number of subtasks to use. Using the MAXLSORT option can improve recovery performance.

Valid values are 0 to 999. If you do not specify a value for MAXLSORT, RECOVER PLUS uses the value of the MAXLSORT installation option (page 553). The MAXLSORT installation option has a default value of 0 (MAXLSORT=0), which allows RECOVER PLUS to determine an appropriate value for MAXLSORT.

If you specify MAXLSORT 1, you set up your job to run as it would in RECOVER PLUS version 8.1.00 and earlier and turn off parallel log sorts and parallel MERGE/SNAP/RESTORE phases. These phases then run serially in the main task in the order in which they are scheduled by the RECOVER PLUS planning component.

The following files are dynamically allocated if you do not code them in JCL:

- **LOGOUnnn**: sort message files

  `nnn` is the number of the log sort and is a number between 1 and the (non-zero) value that is specified for MAXLSORT.

- **LxxxWKnn**: sort work files

  `xxx` is the number of the log sort and is a number between 1 and the (non-zero) value that is specified for MAXLSORT. `nn` is the number of the work data set. For example, if MAXLSORT=3 and two sort work files are required for each sort, the DDs would be specified as follows:
When you use dynamic allocation for these files, RECOVER PLUS determines the optimal number of files to use.

When MAXLSORT is greater than 1, the MERGE/SNAP/RESTORE phases run in subtasks in parallel. The order of execution of phases may be different from the execution plan in AFRPLAN. RECOVER PLUS uses deferred messaging for these phases and phase messages are printed in AFRPRINT in the order in which the phases complete rather than in the order specified in AFRPLAN. The order of message output for phases may not be repeatable from job to job. (As in previous releases, AFRSUMRY messages are printed in execution plan order.)

For more information about setting MAXLSORT and how the parallel merge phase works, see “Setting the MAXLSORT and KSORTSHARE options” on page 441.

**KSORTSHARE**

The KSORTSHARE option specifies if key sorts are shared among RECOVER PLUS table space recoveries (MERGE phases) running in parallel.

If you do not specify a value on the OPTIONS command for KSORTSHARE, RECOVER PLUS uses the value of the KSORTSHARE installation option (page 554), which defaults to YES.

When you specify YES, RECOVER PLUS uses up to the value specified for MAXKSORT for active key sorts at any given time. If sufficient key sorts are not available when a table space recovery begins execution, RECOVER PLUS obtains keys later in an UNLOAD phase.

When you specify NO, each MERGE phase has its own set of key sorts and up to MAXKSORT * MAXLSORT key sorts can be active at any given time. Since the number of sorts that can be active in a system is fairly small – usually no more than 30 – a value of NO for this option may severely limit the number of recovery operations that RECOVER PLUS can perform in parallel when index rebuilds are also requested.

For more information about setting KSORTSHARE and how the parallel merge phase works, see “Setting the MAXLSORT and KSORTSHARE options” on page 441.
**AUTOSIZE**

This option turns dynamic sizing for change accumulation output files on or off. Valid values are YES and NO. If you specify a value for AUTOSIZE, you override the AUTOSIZE installation option, which defaults to YES.

**AUTOSIZE YES**

AUTOSIZE YES specifies that dynamic sizing of change accumulation output files to DASD occurs.

**AUTOSIZE NO**

AUTOSIZE NO specifies that change accumulation output files to DASD are allocated using the primary and secondary quantities that are specified in the R+/CHANGE ACCUM repository.

**RECOVERYPOINT**

The RECOVERYPOINT option supports the Recovery Management for DB2 timestamp recovery feature.

---

**NOTE**

To use a timestamp recovery, you must have the Recovery Management solution, with its valid solution password, installed. Without the solution password, jobs that use RECOVERYPOINT fail.

---

Recovery Management for DB2 can perform a consistent, point-in-time recovery to a user-specified timestamp by using Inflight Resolution technology. RECOVER PLUS translates the timestamp to an RBA or LRSN, recovers the objects, then resolves all inflight units of work for both data sharing and non-data-sharing systems. You can perform timestamp recovery either by using the online interface of RECOVERY MANAGER or by submitting a batch RECOVER PLUS job with the RECOVERYPOINT option. (The online interface of RECOVERY MANAGER for DB2 supports this feature on data sharing systems.)

The Inflight Resolution feature of the Recovery Management for DB2 solution enables you to perform a consistent recovery to any LRSN, RBA, or timestamp. The ability to resolve inflight units of work at any point in time completely eliminates the need to perform quiesces to establish consistent recovery points during application execution. The ability to avoid quiesces can dramatically improve the availability of your DB2 data.

For more information, see the *Recovery Management for DB2 User Guide* and the *RECOVERY MANAGER for DB2 User Guide*.
The RECOVERYPOINT option specifies the recovery point and directs the utility to resolve inflight units of recovery. You specify the RECOVER statements as if a recovery to the current point in time is specified. You can have multiple RECOVER statements in the input. The RECOVERYPOINT option effectively determines what the current recovery point is. You can specify the TIMESTAMP option or the LOGPOINT option (but not both) with RECOVERYPOINT.

In the following examples, the first example shows the syntax with the TIMESTAMP option and the second example shows the syntax with the LOGPOINT option:

```
OPTION RECOVERYPOINT URIDDDN(output) TIMESTAMP yyyy-mm-dd-h.mm.ss.tttttt
OUTPUT(output) UNIT SYSDA
DSNAME dataSetName
RECOVER TABLESPACE databaseName1.tableSpaceName1
    TABLESPACE databaseName2.tableSpaceName2
```

```
OPTION RECOVERYPOINT URIDDDN(output) LOGPOINT X'logPoint'
OUTPUT(output) UNIT SYSDA
DSNAME dataSetName
RECOVER TABLESPACE databaseName1.tableSpaceName1
    TABLESPACE databaseName2.tableSpaceName2
```

You should consider the following items when you use the RECOVERYPOINT option.

- RECOVERYPOINT is not valid for use with LOB table spaces.

- RECOVERYPOINT is not valid with the use of the ACCUM command in the same SYSIN.

- RECOVERYPOINT interacts with the SIMULATION option in the following ways:
  - If OPTION SIMULATE specifies no LOGPOINT, OPTION RECOVERYPOINT determines the ending point of the simulation.
  - If OPTION RECOVERYPOINT specifies a TIMESTAMP, OPTION SIMULATE must not specify a LOGPOINT.
  - If both OPTION RECOVERYPOINT and OPTION SIMULATE specify LOGPOINT, they must both specify the same value.

- If the job includes a RECOVER INDEX command with OPTION INDEXLOG YES, the request fails unless OPTION BACKOUT is also specified. Recovery of an index with RECOVER INDEX with OPTION RECOVERYPOINT is only supported for BACKOUT.

- OPTIONS INDEXLOG AUTO with OPTIONS RECOVERYPOINT converts to a REBUILD unless BACKOUT is also specified.
To use OUTCOPY ONLY and OBIDXLAT with OPTIONS RECOVERYPOINT, you will need to rebuild any indexes on the target table spaces.

If the RECOVERYPOINT specifies a point on the log during a SHRLEVEL CHANGE image copy, forward recovery from that copy fails if the copy includes any changes that were inflight at the recovery point. If that happens, RECOVER PLUS attempts to fall back to a previous copy.

If OPTIONS RECOVERYPOINT is specified, no other point-in-time specification (TORBA, TOLOGPOINT) is allowed.

**URIDDDN (output)**

You use the URIDDDN parameter to identify an OUTPUT statement that is used to allocate a data set where inflight units of recovery are reported.

The URIDDDN parameter refers to an OUTPUT statement to capture a UNIT name and data set name (DSNAME). If you specify the SPACE parameters, RECOVER PLUS also uses them. If the OUTPUT statement specifies a tape unit, RECOVER PLUS ignores it. RECOVER PLUS honors any SMS class specifications (DATACLAS, MGMTCLAS, STORECLAS) on the OUTPUT statement. RECOVER PLUS ignores any other OUTPUT statement parameters.

If the RECOVERYPOINT request does not include a URIDDDN specification, RECOVER PLUS defaults the data set name to:

```
&SYSUID..D&DATE..T&TIME.URIDS
```

An example data set name that uses this default is RDAMSM.D110206.T171005.URIDS.

If the DSNAME parameter on the OUTPUT statement is a relative GDG name, RECOVER PLUS resolves it to an absolute name before allocation.

If the request does not include a URIDDDN specification or if the URIDDDN does not include UNIT, RECOVER PLUS uses the WKUNIT value defined in the installation options.

If the request does not include a URIDDDN specification or if the URIDDDN does not include SPACE parameters, RECOVER PLUS defaults the space to TRK(1,1).

---

**NOTE**

RECOVER PLUS deletes this data set at the end of a successful job.
TIMESTAMP yyyy-mm-dd-h.mm.ss.tttttt

The TIMESTAMP option specifies the recovery point. If you use TIMESTAMP, you cannot use LOGPOINT. TIMESTAMP requires a value to at least the second. You need to specify unspecified levels of detail with zeros. For example, a TIMESTAMP specification that only goes to the minute level needs to specify zeros for the seconds level.

The TIMESTAMP value is converted to a log point and adjusted for Daylight Savings Time. The Daylight Savings Time adjustment uses the TZRULE installation option. For information about this installation option, see page 548.

LOGPOINT X’logPoint’

The LOGPOINT X’logPoint’ option specifies the recovery point for the recovery.

LOGPOINT LOGMARK logMarkName (logMarkGeneration)

The LOGPOINT LOGMARK logMarkName option specifies the recovery point as a log mark that was created in Log Master.

Optionally, you can add a log mark generation number in parentheses, (logMarkGeneration). If you do not specify a log mark generation, RECOVER PLUS uses the most recent version of the log mark. You can use one of the following ways to specify the log mark generation:

- If you specify a log mark generation as less than or equal to zero (<= 0), RECOVER PLUS treats the generation as a relative generation. RECOVER PLUS refers to the most recent log mark with the generation of zero (0). The previous generation of the log mark is referred to with (-1), and so on.

- If you specify a log mark generation as greater than zero (> 0), RECOVER PLUS treats the generation as an absolute generation number and uses the specified version of the log mark.

NOTE

The way that you specify an absolute generation number for a log mark in RECOVER PLUS is different than the way you specify the absolute generation number in Log Master.
When you create the log mark, you can specify if Log Master should create a quiesce point. You can also specify that Log Master set a log mark at a quiet point or a non-quiet point. If you know that the log mark is not at a quiesce point or a quiet point for the spaces or spaces that you are recovering, you should use a timestamp recovery by using OPTIONS RECOVERYPOINT LOGMARK syntax. If you know that the log mark is at a quiesce point or a quiet point, consider using RECOVER TOLOGPOINT LOGMARK syntax (page 183) to perform a point-in-time recovery.

**NOTE**

You must determine if the log mark is at a quiesce or quiet point or not. If you are not sure that there is a quiesce or quiet point, a timestamp recovery may be best although it will take longer to look for inflight transactions.
OUTPUT

This section describes the syntax of the OUTPUT command and its options.

OUTPUT syntax

Figure 7 shows the syntax of the OUTPUT command. Figure 8, Figure 9 on page 146, and Figure 10 on page 146 show the syntax for the Common Options, Disk Options, and Tape Options shown in Figure 7. For the conventions used in these diagrams, see “Syntax diagrams” on page 24.

Figure 7  OUTPUT syntax

Figure 8  OUTPUT syntax—Common options
OUTPUT command dependencies and prohibitions

When you use an OUTPUT command in RECOVER PLUS, the following rules apply:

- The command statement must start with the OUTPUT command followed by the name of the descriptor that you want to use to dynamically allocate your output data sets.

- You must specify DSNAME.

- You can specify only options that apply to the media that you use. That is, all of the options must apply either to disk data sets or to tape data sets.

- You can have more than one OUTPUT command statement in a SYSIN data set, but each output descriptor must have a different name.
Options that are specific to disk data sets and those specific to tape data sets are mutually exclusive: you cannot specify disk output data sets and tape output data sets in the same OUTPUT command statement. To specify disk and tape output data sets in the same SYSIN data set, you must use one OUTPUT command statement for the disk data sets and another for the tape data sets. The names of the descriptors must be different.

For more information, see “Allocating output image copy data sets dynamically” on page 452.

**OUTPUT option descriptions**

RECOVER PLUS provides options for use with the OUTPUT command. The options are described in the order in which they are shown in the OUTPUT syntax diagram.

**OUTPUT name**

Use OUTPUT name to introduce a new output descriptor name. RECOVER PLUS creates the named descriptor by using the options specified in the OUTPUT command statement. The value for name must not exceed eight characters.

**DSNAME dataSetName**

Specify DSNAME and a data set name (dataSetName) to specify a data set name or model to use.

DSNAME is not required if you use the OUTCOPYDSN keyword (page 208), or the RECOVERYDSN keyword (page 209), or both in the OUTCOPY specification on the RECOVER command statement.

You can construct dataSetName by using any of the symbolic variables in Table 6.

<table>
<thead>
<tr>
<th>Symbolic variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;JOBNAME</td>
<td>JOB name used in the JCL</td>
</tr>
<tr>
<td>&amp;STEPNAME</td>
<td>STEP name used in the JCL (PROC names are ignored)</td>
</tr>
<tr>
<td>&amp;DB</td>
<td>database containing the space</td>
</tr>
<tr>
<td>&amp;TS</td>
<td>table space or index space</td>
</tr>
<tr>
<td>&amp;SP</td>
<td>table space or index space</td>
</tr>
<tr>
<td>&amp;DSNUM or &amp;PART</td>
<td>data set or partition (2 digits for 0 - 99, 3 digits for 100 - 999, 4 digits for 1000 – 4096)</td>
</tr>
</tbody>
</table>
Common options specification description

This section describes options that you can use for output data sets written to disk or tape.

UNIT name

Specify UNIT and a new tape or disk unit name when you want to override an existing default unit. The RECOVER PLUS default is SYSALLDA.

RECOVER PLUS can dynamically determine whether a unit is tape or disk.

Table 6  Symbolic variables used with DSNAME and their descriptions (part 2 of 2)

<table>
<thead>
<tr>
<th>Symbolic variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;LDSNUM or &amp;LPART</td>
<td>data set or partition (3 digits for 000–999 and 4 digits for 1000–4096)</td>
</tr>
<tr>
<td>&amp;USERID or &amp;UID</td>
<td>job or TSO user ID</td>
</tr>
<tr>
<td>&amp;SSID</td>
<td>DB2 subsystem ID</td>
</tr>
<tr>
<td>&amp;ATTACH</td>
<td>DB2 group attachment name or subsystem ID</td>
</tr>
<tr>
<td>&amp;DATE</td>
<td>current date (in the form YYMMDD)</td>
</tr>
<tr>
<td>&amp;TIME</td>
<td>current time (in the form HHMMSS)</td>
</tr>
<tr>
<td>&amp;JDATE</td>
<td>current Julian date (in the form YYDDDD)</td>
</tr>
<tr>
<td>&amp;YEAR</td>
<td>current year (in the form YY)</td>
</tr>
<tr>
<td>&amp;MONTH</td>
<td>current month (in the form MM)</td>
</tr>
<tr>
<td>&amp;DAY</td>
<td>current day (in the form DD)</td>
</tr>
<tr>
<td>&amp;JDAY</td>
<td>current Julian day (in the form DDD)</td>
</tr>
<tr>
<td>&amp;HOUR</td>
<td>current hour (in the form HH)</td>
</tr>
<tr>
<td>&amp;MINUTE</td>
<td>current minute (in the form MM)</td>
</tr>
<tr>
<td>&amp;SECOND</td>
<td>current second (in the form SS)</td>
</tr>
<tr>
<td>&amp;ICTYPE</td>
<td>the type of image copy being produced, F for FULL YES</td>
</tr>
<tr>
<td>&amp;UTIL</td>
<td>utility ID (truncated to eight characters)</td>
</tr>
<tr>
<td>&amp;SEQ</td>
<td>a sequential number that restarts at 1 at the beginning of each job step execution</td>
</tr>
<tr>
<td>&amp;TYPE</td>
<td>type of output being produced: LP for local site primary, LB for local site backup, RP for recovery site primary, RB for recovery site backup</td>
</tr>
<tr>
<td>&amp;INST</td>
<td>instance number, with valid values of 1 or 2</td>
</tr>
</tbody>
</table>

For more information, refer to “GDGs and symbolic variables in data set name construction” on page 454.
**CATLG**

Specify CATLG to indicate whether to redefine the operating system catalog directive for the named descriptor.

**CATLG YES**

CATLG YES is the default and redefines the operating system catalog directive for the named descriptor.

If any SMS option (STORCLAS, DATACLAS, or MGMTCLAS) is used, RECOVER PLUS forces CATLG YES.

**CATLG NO**

CATLG NO does not redefine the operating system catalog directive for the named descriptor.

**MODELDCB dataSetName**

Specify MODELDCB and a cataloged data set name (dataSetName) to redefine the model data control block (DCB) for the named descriptor.

To specify that no model DCB be used, use MODELDCB NONE.

The specified model data set must be allocated on a mounted direct access volume. RECOVER PLUS copies the DCB information from the data set label.

You can construct dataSetName by using symbolic variables. Refer to the DSNAME description on page 147 for a list of the symbolic variables that you can use.

**VOLCNT integer**

To set VOLCNT integer for the named descriptor, specify the largest number of volumes that you expect RECOVER PLUS to process when copying a single data set. For both tape and disk data sets, integer must be equal to or greater than the number of volumes produced for the single largest output copy, whether or not you use stacked output. The default is 25. To use the operating system default, set VOLCNT to 0.

---

**NOTE**

If you are using SMS in your system, BMC recommends that you use VOLCNT 0.
**UNITCNT integer**

Use UNITCNT to specify the unit count used for dynamic allocation. Valid values are 0 (zero) to 59. The value 0, the default, means that the unit count is not specified for the allocation.

Specifying UNITCNT 2 for tape output allocates two tape drives. When a tape volume is at the end of tape, RECOVER PLUS begins writing on the second drive immediately. This eliminates time spent waiting for tape rewind.

The total number of tape drives allowed in use at one time is specified by the MAXDRIVES option, either as coded in the OPTIONS command statement or from the MAXDRIVE installation option. If the value for UNITCNT is greater than one and would cause the total number of tape drives in use to exceed MAXDRIVES, RECOVER PLUS decrements UNITCNT to prevent the allocation from exceeding that limit.

**DATACLAS name**

Specify DATACLAS name when you want to provide an SMS data class name for the named descriptor. The value of name must be a valid SMS data class name, not exceeding eight characters.

RECOVER PLUS forces CATLG=YES when you specify DATACLAS.

**MGMTCLAS name**

Specify MGMTCLAS name when you want to provide an SMS management class name for the named descriptor. The value of name must be a valid SMS management class name, not exceeding eight characters.

RECOVER PLUS forces CATLG=YES when you specify MGMTCLAS.

**STORCLAS name**

Specify STORCLAS name when you want to provide an SMS storage class name for the named descriptor. The value of name must be a valid SMS storage class name, not exceeding eight characters.

RECOVER PLUS forces CATLG=YES when you specify STORCLAS.
Disk options specification description

This section describes options that apply only to output data sets written to disk devices.

**SPACE (primary, secondary) allocation unit**

Specify SPACE to set the output allocation units (tracks or cylinders) for the named descriptor. Specify SPACE TRK to allocate the output in tracks. Specify SPACE CYL to allocate the output in cylinders. The default value for this option is CYL. Specify the values in parentheses, as in the following example: SPACE (200,100) CYL. If no SPACE specification is provided, the default is (5,5) CYL.

**VOLUMES (volume1, volume2, .........., volumen)**

Specify VOLUMES to provide a list of default volumes for the named descriptor. The number of entries in the list must not exceed the value specified by VOLCNT for the named descriptor (see page 149). If the data set is uncataloged, RECOVER PLUS truncates the list recorded in SYSIBM.SYSCOPY to reflect the actual volumes used.

---

**WARNING**

Enough space must exist on the first specified volume to allocate the primary space required for the output data set.

---

**EATTR**

Use EATTR to specify whether a data set supports extended attributes or not. If you do not specify EATTR on the OUTPUT command, RECOVER PLUS uses the value of the EATTR installation option (page 557). The EATTR installation option defaults to EATTR=NONE.

For **z/OS Version 1.11 or later**, supports the EATTR option. For earlier versions of **z/OS**, you must set EATTR=NONE.

You cannot use an image copy made to the cylinder-managed portion of an extended address volume (EAV) under **z/OS Version 1.11 on z/OS Version 1.10** because **z/OS Version 1.10** does not support sequential data sets in the cylinder-managed portion of an EAV.

If you are using **DB2 Version 9 and z/OS Version 1.11**, RECOVER PLUS can create a data set with extended attributes and will register it in SYSCOPY. However, the **DB2 RECOVER Version 10 utility** is required to work with the registered data set.

You can also set EATTR to OPT or NO in the JCL.
When you create an OUTCOPY (page 199) and want to force the copy to the cylinder-managed space of an EAV, specify the SPACE keyword (page 204).

Valid values for EATTR are:

- NONE specifies no value for EATTR and allows the value for EATTR to be set by an SMS DATACLAS

  Using NONE allows you to have your environment set up to use extended attributes.

- OPT specifies that extended attributes are optional for the data set.

  You must set EATTR=OPT to allocate an extended format sequential data set. By using EATTR=OPT, RECOVER PLUS supports sequential data sets in the cylinder-managed portion of EAVs.

  Extended format sequential data sets must be allocated on SMS-managed volumes and the size of the data set must be greater than the EAV break point, which is typically 10 cylinders.

- NO specifies that the data set cannot have extended attributes.

**Tape options specification description**

This section describes the options that apply only to copies written to tape.

**STACK**

The STACK option tells RECOVER PLUS whether to stack the output copies from multiple RECOVER executions contiguously on the same tape volumes. For information about using tape stacking, see “Stacking copies on tape” on page 456.

**STACK NO**

STACK NO, the default, tells RECOVER PLUS not to stack output copies contiguously on tape.
STACK YES

STACK YES tells RECOVER PLUS to stack output copy data sets (of the same type) from multiple RECOVER command statements contiguously on the same tape.

WARNING

If you are using Tape Mount Management (TMM), be aware that TMM intercepts any data set allocation whether dynamic or otherwise. If you want the copies on tape and use STACK YES with TMM, add the RECOVER PLUS program AFRMAIN to the TMM exclusion list.

If RECOVER PLUS detects that the allocation has gone to disk instead of tape, it discontinues stacking.

TRTCH

The TRTCH option allows the use of Improved Data Recording Capability (IDRC) tape compression for the named descriptor.

TRTCH NONE

This value is the default and specifies the use of the system default for compression. TRTCH NONE is equivalent to not specifying TRTCH.

TRTCH COMP

TRTCH COMP specifies that IDRC compression is to be used for output files.

TRTCH NOCOMP

TRTCH NOCOMP specifies no IDRC compression for output files (even if the system default is TRTCH COMP).

RETPD integer

Specify RETPD integer to set a new retention period (in days) for the current output data set. The value of integer must be in the range of 1 through 9999.

NOTE

When EXPDT is specified, it takes precedence over RETPD.
**EXPDT date**

Specify EXPDT `date` to set a new expiration date for the current output data set. The default value of the EXPDT option is 99000, which indicates no expiration date.

---

**NOTE**

When it is specified, EXPDT takes precedence over RETPD.

---

The value of `date` must be in the format `YYDDD`, `YYYYDDD`, or `YYYY/DDD`, where `YYYY` is the 4-digit year, `YY` is the last two digits of the year, and `DDD` is the 3-digit Julian day (001 through 366).

---

**NOTE**

A date with a two-digit year is passed as is to dynamic allocation. For years beyond 1999, depending on your environment, this date may not produce the desired result. BMC recommends using a four-digit year.
This section describes the syntax of the RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE commands and their options.

**RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax**

Figure 11 through Figure 25 on page 163 show the syntax of the RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE commands. For the conventions used in these diagrams, see “Syntax diagrams” on page 24.
RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax

Figure 12  RECOVER syntax—Table space specification

---

**Table space specification**

- **TABLESPACE**
  - tableSpaceName
    - DSNDB04
    - databaseName
  - OBJECTSET
    - creatorName
    - page 168
  - INDEXES
    - INDEX
      - NO
      - YES
    - page 168
  - DROPRECOVERY **
    - page 170
    - page 193
  - OBIDXLAT
    - specification
    - page 169
    - page 170
    - page 171
  - FROMRBA or FROMLOGPOINT
    - specification
    - page 160
  - OUTCOPY
    - specification
    - page 161
  - Non-registered copy or INLOG
    - specification
    - page 160
  - TRANSFORM
    - page 171
  - page 160
  - page 161
  - page 161

*Not valid with TRANSFORM
**Not valid with BACKOUT
Figure 13  RECOVER syntax—Single index specification

*For special considerations, see the description on page 171.
** Valid with INDEXLOG AUTO.
Figure 14  RECOVER syntax—Multiple index specification

```plaintext
INDEX (indexName)
  creatorID.
  TABLESPACE DSNDB04.
databaseName.

INDEXSPACE (indexSpaceName)
  TABLESPACE DSNDB04.
databaseName.

NOWORKDDN***
```

**Allowed with multiple recoveries only if dynamic outcopy allocation (OUTPUT syntax is used.
**For special considerations, see the description on page 174.
*** Valid with INDEXLOG AUTO.

Figure 15  RECOVER syntax—INDEPENDENT OUTSPACE specification

```
INDEPENDENT OUTSPACE specification

INDEP OUTSPACE* (vcat.BMCDBC.databaseName.objectName.I0001_znnn)
  MODEL
  userNamedDataSet

*Not valid with BACKOUT
```
Figure 16  RECOVER syntax—Point-in-time recovery specification

```
<table>
<thead>
<tr>
<th>Point-in-time recovery specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>TORBA</td>
</tr>
<tr>
<td>TOLOGPOINT</td>
</tr>
<tr>
<td>page 181</td>
</tr>
<tr>
<td>LASTQUIESCE</td>
</tr>
<tr>
<td>(relativeGenerationNumber)</td>
</tr>
<tr>
<td>BACKOUT</td>
</tr>
<tr>
<td>YES</td>
</tr>
<tr>
<td>LASTCOMMONQ</td>
</tr>
<tr>
<td>LASTARCHQ</td>
</tr>
<tr>
<td>LASTSHUTDOWN</td>
</tr>
<tr>
<td>LOGMARK logMarkName</td>
</tr>
<tr>
<td>(logMarkGeneration)</td>
</tr>
<tr>
<td>X'logPoint'</td>
</tr>
</tbody>
</table>
```

Figure 17  RECOVER syntax—TOCOPY specification

```
<table>
<thead>
<tr>
<th>TOCOPY specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOCOPY</td>
</tr>
<tr>
<td>page 185</td>
</tr>
<tr>
<td>LASTCOPY</td>
</tr>
<tr>
<td>(relativeGenerationNumber)</td>
</tr>
<tr>
<td>dataSetName</td>
</tr>
<tr>
<td>TOVOLUME</td>
</tr>
<tr>
<td>CATALOG</td>
</tr>
<tr>
<td>volumeSerialNumber</td>
</tr>
<tr>
<td>TOSEQNO integer</td>
</tr>
</tbody>
</table>
```

Figure 18  RECOVER syntax—LOGSORT specification

```
<table>
<thead>
<tr>
<th>LOGSORT specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGSORT</td>
</tr>
<tr>
<td>page 113</td>
</tr>
<tr>
<td>SORTDEVT deviceType</td>
</tr>
<tr>
<td>page 114</td>
</tr>
<tr>
<td>SORTNUM integer</td>
</tr>
<tr>
<td>page 114</td>
</tr>
<tr>
<td>NUMREC</td>
</tr>
<tr>
<td>page 114</td>
</tr>
<tr>
<td>CALC</td>
</tr>
<tr>
<td>NOEST</td>
</tr>
<tr>
<td>EST integer</td>
</tr>
<tr>
<td>AVGRECSZ integer</td>
</tr>
<tr>
<td>ABS integer</td>
</tr>
<tr>
<td>AVGRECSZ integer</td>
</tr>
</tbody>
</table>
```
RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax

Figure 19  RECOVER syntax—OBIDXLAT specification

Figure 20  RECOVER syntax—Non-registered copy or INLOG specification

*Not valid with BACKOUT

*Not valid with index specification

**Not valid with BACKOUT
RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax

Figure 21  RECOVER syntax—FROMRBA or FROMLOGPOINT specification

FROMRBA or FROMLOGPOINT specification

FROMRBA  FROMLOGPOINT

LASTQUIESCE  LASTCOPY  LASTARCHQ  LASTSHUTDOWN

X'logPoint'

FROMRBA  FROMLOGPOINT

Figure 22  RECOVER syntax—OUTCOPY specification

OUTCOPY specification

OUTCOPY

NO  YES

REGISTER  ALL  NONE

OUTCOPYDDN  RECOVERYDDN

SPACE

CYL

TRK

** Not valid with BACKOUT or Instant Snapshot copies
Figure 23  RECOVER syntax—INCOPY specification

**INCOPY specification**

- **DSNAME** dataSetName  
  - page 204
- **MODEL** dataSetName  
  - page 205
- **DSSIDZ (integerG)**  
  - page 205
- **PIECESIZE (integer)**  
  - page 205
  - integerK
  - integerM
  - integerG
- **RBA**
- **X’logPoint’**  
  - page 206
- **SHRLEVEL**  
  - page 206
  - CHANGE
- **REFERENC**  
  - page 206
  - CHANGE
- **ENCRYPTED**  
  - page 206
- **TIMESTAMP**  
  - page 206
- **COMPRESSED**  
  - page 207
- **INVOLUME**  
  - page 207
- **CATALOG**  
  - page 207
  - volumeSerialNumber
  - INDEVT deviceType
  - INSEQNO integer

*Valid only with INCOPY FULL SNAPSHOT
**Not valid with table space specification

Figure 24  RECOVER syntax—OUTCOPYDDN specification

**OUTCOPYDDN specification**

- **OUTCOPYDDN**  
  - page 208
- **DATA**  
  - BMCCPY**  
  - DDName
  - DDNamePrefix
  - outputDescriptor1

- **OUTCOPYDSN**  
  - page 208
  - dataSetName1
  - dataSetName2

*These default values apply to table spaces with fewer than 100 partitions. If the table space has 100 partitions or more, the corresponding default values are BMCCY and BMCCZ.
**These values are the default values that are used by the OUTPUT command.
***These values are used to specify the OUTPUT command, which will control the allocation of the copy.
RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax

Figure 25  RECOVER syntax—RECOVERYDDN specification

<table>
<thead>
<tr>
<th>RECOVERYDDN specification</th>
<th>RECOVERYDSN - ( dataSetName3</th>
</tr>
</thead>
<tbody>
<tr>
<td>( BMCRCY***</td>
<td>outputDescriptor3***</td>
</tr>
<tr>
<td>DDName3</td>
<td>dataSetName4</td>
</tr>
<tr>
<td>DDNamePrefix3</td>
<td>)</td>
</tr>
<tr>
<td>outputDescriptor3***</td>
<td>)</td>
</tr>
<tr>
<td>BMCRCZ***</td>
<td>dataSetName4</td>
</tr>
<tr>
<td>DDName4</td>
<td>)</td>
</tr>
<tr>
<td>DDNamePrefix4</td>
<td>)</td>
</tr>
<tr>
<td>outputDescriptor4***</td>
<td>)</td>
</tr>
</tbody>
</table>

*These default values apply to table spaces with fewer than 100 partitions. If the table space has 100 partitions or more, the corresponding default values are BMCRY and BMCRZ.

**These values are the default values used by the OUTPUT command.

***These values are used to specify the OUTPUT command, which will control the allocation of the copy.
RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX command dependencies and prohibitions

Table 7 describes the common dependencies for coding the RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX commands.

<table>
<thead>
<tr>
<th>Option coded</th>
<th>Option that must be used</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCOPY</td>
<td>TOCOPY LASTCOPY OR TOCOPY data set name where data set name matches the last DSNAME in INCOPY OR RBA/LOGPOINT on the last INCOPY specification</td>
</tr>
<tr>
<td>OBIDXLAT and INCOPY</td>
<td>TOCOPY on the RECOVER TABLESPACE command statements OR supply the source DBID and PSID in the OBIDXLAT clause</td>
</tr>
<tr>
<td>FROMRBA or FROMLOGPOINT</td>
<td>LOGAPPLY ONLY</td>
</tr>
<tr>
<td>BACKOUT</td>
<td>TORBA or TOLOGPOINT</td>
</tr>
</tbody>
</table>

Table 8 describes the common prohibitions to coding the RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX commands.

<table>
<thead>
<tr>
<th>Option coded</th>
<th>Option that must not be used</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOCOPY</td>
<td>LOGAPPLY ONLY OR LOGONLY</td>
</tr>
<tr>
<td>INCOPY or INLOG</td>
<td>LOGAPPLY ONLY OR LOGONLY</td>
</tr>
<tr>
<td>INCOPY and OBIDXLAT without specifying the source IDs and do NOT code DROPRECOVERY</td>
<td>RBA or LOGPOINT on any INCOPY specification</td>
</tr>
<tr>
<td>DROPRECOVERY</td>
<td>INDEP OUTSPACE</td>
</tr>
<tr>
<td>BACKOUT</td>
<td>INDEP OUTSPACE, OUTCOPY ONLY, OBIDXLAT, or DROPRECOVERY</td>
</tr>
</tbody>
</table>
RECOVER INDEX and RECOVER INDEXSPACE command dependencies and prohibitions

The same dependencies and prohibitions that are listed in Table 7 on page 164 and in Table 8 on page 164 for the RECOVER TABLESPACE command also apply to the RECOVER INDEX and RECOVER INDEXSPACE commands. Additionally, you should consider the following items when you recover indexes.

**NOTE**
An index space does not have to be defined with the COPY YES attribute to be recoverable.

- If you have used the BMC PACLOG for DB2 utility to exclude the index log records, index recovery from logs is not possible.

- If you specify RECOVER INDEX (ALL) or RECOVER INDEXSPACE (ALL), RECOVER INDEX and RECOVER INDEXSPACE do not support the FROMRBA/FROMLOGPOINT options and only support the OUTCOPY option when the copies are dynamically allocated. (For syntax regarding dynamic allocation, see “OUTPUT” on page 145.) You can use separate index specifications with RECOVER INDEX and RECOVER INDEXSPACE to use those options.

- If you have used an ALTER statement to change the PIECESIZE for an index, index recovery from log is not possible.

- If a table space is recovered to a prior point in time, the default TORBA/TOLOGPOINT value for recovery of any indexes on that table space is the log point specified for the table space. If you specify different log points for the table space and its index, either with TORBA/TOLOGPOINT or TOCOPY, RECOVER PLUS ends with an error.

- If you recover a table space by using the TOCOPY and the INCOPY keywords, and you also recover a related index by using the TOCOPY and the INCOPY keywords, RECOVER PLUS cannot verify that the two copies are in sync.

- RECOVER PLUS does not prevent a point-in-time recovery of an index that does not include a point-in-time recovery of the table space or other indexes on the table space.

- Indexes must always have a copy or other backup if RECOVER INDEX or RECOVER INDEXSPACE is used without BACKOUT. If an index is created on a table that contains data, DB2 does not log the index updates. Similarly, index rebuilds or updates that result from LOG YES utilities are not logged. If the index is created on a table, an image copy or other backup must be made before the index is recoverable.
In addition to the LOG NO SYSCOPY events that make a table space unrecoverable, any LOAD LOG YES event or REORG LOG YES event between the image copy and the TORBA/TOLOGPOINT value makes an index unrecoverable. REBUILD INDEX and REORG INDEX command statements can also render an index unrecoverable, and RECOVER PLUS cannot detect these events for COPY NO indexes if non-BMC utilities are used.

RECOVER PLUS REBUILD INDEX events are registered in the BMCXCOPY table for all COPY NO indexes. BMC REORG PLUS registers REORG INDEX events for COPY NO indexes in the BMCXCOPY table.

You cannot use RECOVER INDEX or RECOVER INDEXSPACE commands for forward index recovery through a point-in-time recovery event created by a BACKOUT job.

If you specify the INCOPY option with OBIDXLAT, and you do not specify INDEP OUTSPACE or OUTCOPY, a warning message stating that a copy is required is issued and the original status of the index and its table space is not restored.

When you use RECOVER INDEXSPACE with OBIDXLAT for indexes, you must also specify the table ID that is associated with the index as an additional OBID(x,y) clause, where x is the ID of the source table and y is the ID of the target table. If you do not specify this extra OBID clause, the job issues the following message:

```
BMC40764I TABLE OBID X’xxxx’ FOUND AT OFFSET X’3A’ IN INDEX HEADER
WILL NOT BE TRANSLATED
```

You may, however, omit both OBID(x,y) clauses. If you omit both clauses, appropriate default values are supplied. For more information, see “OBIDXLAT specification” on page 193.
RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE option descriptions

This section describes the options available with RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE.

Not all of the options described are valid in a single RECOVER TABLESPACE, RECOVER INDEX, or RECOVER INDEXSPACE command. The recovery strategy that best meets your needs determines which options you should include in a single command statement.

RECOVER

Use the RECOVER command when you want to perform an actual recovery.

SIMRCVR

SIMRCVR is a synonym for the RECOVER command. Use the SIMRCVR command to clarify that you are running in simulation mode. When you use SIMRCVR, you must also specify OPTIONS SIMULATE YES, which requires a valid Recovery Management for DB2 solution password. For more information, see “SIMULATE YES” on page 134.

The SIMRCVR command generally has the same meaning and the same syntax in simulation mode as a regular RECOVER command in non-simulation mode. However, the following limitations apply to SIMRCVR:

- You cannot use any of the following options:
  
  — BACKOUT 
  — LOGONLY 
  — LOGAPPLY ONLY 
  — TOCOPY 
  — TOLOGPOINT 
  — TORBA 

  **NOTE**

  Even though you cannot specify TOLOGPOINT or TORBA on the SIMRCVR command statement, if you are using the Recovery Management for DB2 solution and have a valid password for the solution, you can simulate a recovery to a previous point in time by coding SIMULATE YES LOGPOINT X’logPoint’ on the OPTIONS command statement.

- You cannot use the INLINE or SNAPSHOT option in the INCOPY specification.
Table space specification

The syntax diagram for the Table Space Specification is in Figure 12 on page 156. For option descriptions for specifications which are included in the Table Space Specification, see “OBIDXLAT specification” on page 193, “Non-registered copy or INLOG specification” on page 195, “FROMRBA or FROMLOGPOINT specification” on page 198, and “OUTCOPY specification” on page 199.

**TABLESPACE databaseName.tableSpaceName**

The TABLESPACE option specifies the table space to be recovered.

- *databaseName* is the name of the database containing the table space. The default is DSNDB04. The name cannot be DSNDB01, DSNDB06, DSNDB07, or any database that is defined with TYPE='W' in SYSIBM.SYSDATABASE.

- *tableSpaceName* is the name of the table space within the named database. BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.

**OBJECTSET**

Use the OBJECTSET option with TABLESPACE to recover only the table spaces that are in a defined group. For more information, see “OBJECTSET specification” on page 176 and “Using BMC RECOVERY MANAGER groups” on page 476.

---

**NOTE**

To include the indexes in a recovery using TABLESPACE OBJECTSET, specify INDEXES YES (page 169).

## INDEXES

The INDEXES option allows you to specify that you want RECOVER PLUS to recover the indexes associated with the table space(s) given by the TABLESPACE option of the RECOVER command. The default is INDEXES NO indicating no indexes recovery.

---

**NOTE**

The use of INDEX is synonymous to INDEXES for this option.

The INDEXES option is not applicable to INDEXSPACE or INDEX specifications.
INDEXES NO

Specifying INDEXES NO tells RECOVER PLUS to not recover the indexes for the specified table space or table spaces.

INDEXES YES

Specifying INDEXES YES tells RECOVER PLUS to recover all indexes associated with the table space(s) specified by the TABLESPACE.

NOTE

INDEXES YES is invalid with an unqualified OBJECTSET specification (OBJECTSET without TABLESPACE).

DSNUM

DSNUM specifies a single data set in the specified table space or the entire table space for recovery.

DSNUM ALL

DSNUM ALL is the default and specifies that all of the data sets in the table space are to be recovered. If the table space is nonpartitioned and DSNUM ALL is specified or implied, image copies made for specific data sets will only be considered for recovery if they are registered at the same log point.

DSNUM integer

Use this option to recover only one partition or data set in the table space. For partitioned table spaces, DSNUM integer specifies the number (from 1 through 4096) of a single partition in the specified table space. For nonpartitioned table spaces, DSNUM integer specifies the number (from 1 to 32) of a single data set of the specified table space.

DSNUM begin : end

Use this option to recover a range of partitions in the table space. For partitioned table spaces, begin specifies the number (from 1 through 4095) of the first partition in the range and end specifies the number (from 2 through 4096) of the last partition in the range. The two numbers are separated by a colon (:) with or without spaces. Wrapping partition numbers (for example, DSNUM 4050 : 300) is not supported.

AUX

The AUX option specifies if auxiliary objects will be included with the recovery of the base table spaces. For more information, see “AUX” on page 122.
DROPRECOVERY

DROPRECOVERY specifies that you want to recover a dropped table space. The DROPRECOVERY keyword requires either INCOPY, INLOG, LOGONLY, or LOGAPPLY ONLY. When DROPRECOVERY is coded, the utility attempts to locate the point at which the table space was re-created and uses that point as a stopping point or verifies it against the TORBA or TOLOGPOINT value. If the space was dropped and re-created several times, you must specify a TORBA/TOLOGPOINT value in the RECOVER PLUS job to tell RECOVER PLUS when to stop applying log records. Use DSN1LOGP to identify the log point of the first CREATE TABLESPACE statement after the drop. This is the log point that you should specify with the TORBA/TOLOGPOINT option. If you do not specify a TORBA/TOLOGPOINT value in this case, the recovery will result in an empty table space.

If the dropped space was versioned, you must recreate the space with the same versions, or in some cases, you can use DB2 REPAIR VERSIONS.

- If the dropped space has been reorganized after the first alter, you can recreate the space to match the latest version of the space, recover the space, and then run DB2 REPAIR VERSIONS. You cannot rebuild indexes until after you run DB2 REPAIR VERSIONS.

- If the dropped space has not been reorganized, you must recreate the space as it existed originally, before any alters, perform the alters to create the same versions that the dropped space had, and then recover the space.

WARNING

Use DROPRECOVERY only if the table space has been dropped (not just a table within the table space).

NOTE

DROPRECOVERY is not valid for index recovery.

For examples on how to use DROPRECOVERY, see Chapter 7, “Recovering a dropped table space or table.”
TRANSFORM

Use TRANSFORM to specify that you want to do a DDL transformation. A TOCOPY LASTCOPY and OBID translation is assumed.

**NOTE**

TRANSFORM is part of the High-speed Structure Change process. SHRLEVEL CHANGE TRANSFORM requires a Recovery Management solution password, but a SHRLEVEL REFERENCE TRANSFORM does not.

The High-speed Structure Change process allows you to perform the following transformations:

- simple table space to partition by growth (PBG)
- non-LARGE partitioned table space to LARGE partitioned table space
- partitioned by range (PBR) table space to universal table space (UTS) PBR
- partitioned by range table space to UTS PBG

The High-speed Structure Change process also allows you to transform your indexes, avoiding costly sort CPU that would be required to rebuild the index.

At a more granular level, you can use the High-speed Structure Change process to simply change the segment size of a table space. This includes transforming a non-segmented table space into a segmented one. You can also change the data set size of your table space or index. This is helpful if you are reaching a capacity limit, i.e., exceeding the data set size or the number of data sets.

See the *Recovery Management for DB2 User Guide* for more information.

**Single index specification**

The syntax diagram for the Single Index Specification is in Figure 13 on page 157. For option descriptions for the OBIDXLAT, Non-Registered Copy, FROMRBA, and OUTCOPY specifications, which are included in the Single Index Specification, see “OBIDXLAT specification” on page 193, “Non-registered copy or INLOG specification” on page 195, “FROMRBA or FROMLOGPOINT specification” on page 198, and “OUTCOPY specification” on page 199.

**NOTE**

If INDEXLOG=NO is specified as an installation option or on the OPTIONS command, single index specifications are not valid on the same RECOVER statement with TABLESPACE. Without TABLESPACE, the parser understands that REBUILD is implied.
**INDEX creatorID.indexName**

The INDEX option specifies the index to be recovered.

- `creatorID` is the qualifier creator ID for the index. The default is the user identifier for the utility.

- `indexName` is the name of the index. Indexes for BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.

**OBJECTSET creator.name**

Use the OBJECTSET option with INDEX to recover only the indexes that are in a defined group. For more information, see “OBJECTSET specification” on page 176 and “Using BMC RECOVERY MANAGER groups” on page 476.

**INDEXSPACE databaseName.indexSpaceName**

The INDEXSPACE option specifies the index space to be recovered.

- `databaseName` is the name of the database containing the index space. The default is DSNDB04. The name cannot be DSNDB01, DSNDB06, DSNDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE.

- `indexSpaceName` is the name of the index space within the named database. Indexes for BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.

**DSNUM**

DSNUM specifies either a single data set for the specified index or the entire index for recovery.

---

**NOTE**

If you code INDEXLOG=NO in the installation options or on the OPTIONS command, RECOVER INDEX is analogous to REBUILD INDEX and you should use PART (page 221) rather than DSNUM. Otherwise, RECOVER PLUS ends, issues BMC40442 and BMC40443, and requests validation of the options.
DSNUM ALL

DSNUM ALL is the default and specifies that all of the data sets for the index are to be recovered.

If the index is nonpartitioned and DSNUM ALL is specified or implied, only image copies made for the entire index (those registered with DSNUM 0) are used for recovery, with the following exception: If DSNUM ALL is specified and RECOVER PLUS detects that one or more DSNUM n copies are registered for an index at the same RBA, RECOVER PLUS expands the request from DSNUM ALL to DSNUM n, (where n is the number of image copies registered together) and issues an informational message.

For multi-data-set, nonpartitioned index spaces, if the index copy used for recovery is a DSN1COPY, DSNUM ALL is not allowed.

DSNUM integer

Use this option to recover only one partition or data set for an index. For partitioned indexes, DSNUM integer specifies the number (from 1 through 4096) of a single partition of the specified index. For nonpartitioned indexes, DSNUM integer specifies the number (from 1 to 4096) of a single data set for the specified index.

DSNUM begin : end

Use this option to recover a range of partitions in the table space. For partitioned table spaces, begin specifies the number (from 1 through 4095) of the first partition in the range and end specifies the number (from 2 through 4096) of the last partition in the range. The two numbers are separated by a colon (:) with or without spaces. Wrapping partition numbers (for example, DSNUM 4050 : 300) is not supported.

NOWORKDDN

The NOWORKDDN option causes the keys to be sent directly to the sort routine without first writing these keys to a work data set. With RECOVER INDEX, NOWORKDDN is only valid when you have also specified INDEXLOG AUTO.
Multiple index specification

The syntax diagram for the Multiple Index Specification is in Figure 14 on page 158. For option descriptions for the OUTCOPY specification, which is included in the Multiple Index Specification, see “OUTCOPY specification” on page 199.

**NOTE**

If INDEXLOG=NO is specified as an installation option or on the OPTIONS command, multiple index specifications are not valid on the same RECOVER statement with TABLESPACE. Without TABLESPACE, the parser understands that REBUILD is implied.

**INDEX creatorID.indexName**

The INDEX option specifies the index to be recovered as follows:

- **creatorID** is the qualifier creator ID for the index. The default is the user identifier for the utility.
- **indexName** is the name of the index. Indexes for BMCUTIL, BMCSYNC, and BMXCCOPY (tables that are used by BMC utilities) cannot be used.

**OBJECTSET creator.name**

Use the OBJECTSET option with INDEX to recover only the indexes that are in a defined group. For more information, see “OBJECTSET specification” on page 176 and “Using BMC RECOVERY MANAGER groups” on page 476.

**INDEX (ALL) TABLESPACE databaseName.tableSpaceName**

(ALL) TABLESPACE specifies that all indexes for the named table space are to be recovered. The parentheses around ALL are optional.

TABLESPACE specifies the table space from which all indexes are to be recovered as follows:

- **databaseName** is the name of the database containing the table space. The default is DSND04. The name cannot be DSND01, DSND06, DSND07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE.

- **tableSpaceName** is the name of the table space within the named database. BMCUTIL, BMCSYNC, and BMXCCOPY (tables that are used by BMC utilities) cannot be used.
AUX

The AUX option specifies if auxiliary objects will be included with the recovery of the base table spaces. For more information, see “AUX” on page 122.

INDEXSPACE databaseName.indexSpaceName

The INDEXSPACE option specifies indexes to be recovered as follows:

- `databaseName` is the name of the database containing the index space. The default is DSNDB04. The name cannot be DSNDB01, DSNDB06, DSNDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE.

- `indexSpaceName` is the name of the index space within the named database. Indexes for BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.

INDEXSPACE (ALL) TABLESPACE databaseName.tableSpaceName

(ALL) TABLESPACE specifies that all indexes for the named table space are to be recovered. The parentheses around ALL are optional.

TABLESPACE specifies the table space from which all indexes are to be recovered as follows:

- `databaseName` is the name of the database to which the table space belongs. The name cannot be DSNDB01, DSNDB06, DSNDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE. The default is DSNDB04.

- `tableSpaceName` is the name of the table space whose indexes are to be recovered. BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.

If you do not specify the TABLESPACE option, RECOVER PLUS determines the table space for the first valid index specified. All other specified indexes must belong to the same table space.

NOWORKDDN

The NOWORKDDN option causes the keys to be sent directly to the sort routine without first writing these keys to a work data set. With RECOVER INDEX, NOWORKDDN is only valid when you have also specified INDEXLOG AUTO.
**OBJECTSET specification**

The OBJECTSET option is available in RECOVER statement syntax to recover groups that are defined using RECOVERY MANAGER. OBJECTSET is available in the following syntactical forms:

- **RECOVER OBJECTSET creator.name**

  This form is shown in Figure 11 on page 155. RECOVER OBJECTSET recovers all objects in a group, both table spaces and indexes.

  _NOTE_

  INDEXES YES is not valid with a RECOVER OBJECTSET specification.

- **RECOVER TABLESPACE OBJECTSET creator.name**

  This form is shown in Figure 12 on page 156. RECOVER TABLESPACE OBJECTSET recovers only the table spaces in the group.

- **RECOVER TABLESPACE OBJECTSET creator.name INDEXES YES**

  RECOVER TABLESPACE OBJECTSET... INDEXES YES recovers the table spaces in the group along with their associated indexes.

- **RECOVER INDEX OBJECTSET creator.name**

  This form is shown in Figure 13 on page 157. RECOVER TABLESPACE OBJECTSET recovers only the indexes in the group.

  _NOTE_

  SIMRCVR is also valid in place of RECOVER.

The following rules apply to `creator.name`:

- `creator` specifies the name of the creator of the group and can have up to 128 characters in length.

- `name` specifies the group name and can have up to 128 characters in length.

- You can delimit both `creator` and `name` with single or double quotation marks.

- Both `creator` and `name` can contain the special characters $, #, and @ in any position.
RECOVER PLUS uses dynamic grouping to determine the group contents at the time that you run the RECOVER PLUS job containing the OBJECTSET option. Because RECOVER PLUS reads the objects in the group each time the job is executed, objects may be added or removed from the group.

**NOTE**

RECOVER PLUS does not read RECOVERY MANAGER recovery options for the group. If RECOVERY MANAGER recovery options change, you must run RECOVERY MANAGER to pick up the new options and generate the control cards.

Following are restrictions that apply when you use OBJECTSET in any of its forms on the RECOVER statement:

- AUX is ignored with any specification of OBJECTSET.

- You cannot specify DSNUM with OBJECTSET. RECOVER PLUS uses the group definition for the object.

- You cannot specify OBIDXLAT with OBJECTSET except for single table table spaces and indexes with no OBIDs specified.

- Only TOCOPY LASTCOPY is valid with OBJECTSET. (You cannot use a specific data set name.)

- You cannot use INDEP OUTSPACE with OBJECTSET.

- You cannot use INCOPY or INLOG with OBJECTSET.

- RECOVER PLUS allows only one OBJECTSET clause, and the statement can not contain additional TABLESPACE, INDEX, or INDEXSPACE clauses.

When you use OBJECTSET, if RECOVER PLUS tries to recover a group of objects and more than 10 of the objects are unrecoverable, RECOVER PLUS issues an error for each object that is unrecoverable and discontinues further processing of the group. This behavior is based on the ERRCONT option that has a default value of 10. To avoid this behavior, set the ON ERROR CONTINUE option with a value that will exceed the possible number of errors in the group, and the recovery will complete.
**INDEPENDENT OUTSPACE specification**

The syntax diagram for the INDEPENDENT OUTSPACE Specification is in Figure 15 on page 158.

**INDEPENDENT OUTSPACE or INDEP OUTSPACE**

Use the INDEPENDENT (or INDEP) OUTSPACE option to redirect the output of the recovery to a data set that is not used by DB2. INDEP OUTSPACE recoveries do not require you to stop the spaces that are being recovered, and they do not interfere with production processing. INDEP OUTSPACE enables you to do the following things:

- test a recovery process without taking spaces offline
- migrate data to another DB2 system
- recover an image to another data set while allowing access to data that is experiencing intermittent errors

For DB2 objects that have multiple underlying VSAM data sets, the following rules apply:

- If the object being recovered is specified by using DSNUM ALL (either explicitly or by default), the output of the recovery is redirected to multiple independent data sets.
- If only a single data set or partition of an object is being recovered (by using DSNUM integer), the output of the recovery is redirected to a single independent data set.

You can specify the name of each independent data set or you can accept the RECOVER PLUS model as a default. (For a diagram of the syntax, see Figure 15 on page 158.)

You can preallocate the INDEP OUTSPACE data sets using the following guidelines:

- For VCAT-defined spaces, ensure that the volumes or space allocated is large enough to hold the data set.
- For DB2 Version 8 or later, if the source space is defined using a STOGROUP, and if primary and secondary quantities are not defined for the source space, RECOVER PLUS uses a sliding-scale calculation for secondary extents, similar to the method DB2 uses.

For more information, see “Primary space allocation” on page 93 and “Secondary extents” on page 94.
You can allow RECOVER PLUS to dynamically allocate the INDEP OUTSPACE data sets for STOGROUP-defined spaces. If the INDEP OUTSPACE data set does not exist, RECOVER PLUS uses the attributes of the source space to dynamically allocate the target INDEP OUTSPACE data set. If the INDEP OUTSPACE data set exists, RECOVER PLUS uses it as is (no DELETE or DEFINE executed).

RECOVER PLUS includes support for multi-data-set spaces. For example, RECOVER PLUS will dynamically allocate A002, A003, to A00n INDEP OUTSPACE data sets as needed.

To have RECOVER PLUS dynamically allocate the INDEP OUTSPACE data set on a specific volume or volumes, pass a volume list by using the STOGROUP...USEORDER specification in the OPTIONS statement (see page 117). RECOVER PLUS statements that use STOGROUP...USEORDER are shown in the following example:

```
OPTIONS STOGROUP TESTSTO USEORDER(NEW004,NEW005,NEW006)  
RECOVER TABLESPACE TESTDB.TESTTS INDEP OUTSPACE
```

The default model name is identical to the DB2 name of the object being recovered, but with BMCDBC substituted for DSNDDBC in the CLUSTER portion of the VSAM data set:

```
vcat.BMCDBC.databaseName.objectName.p0001.znnn
```

The variables are defined as follows:

- `vcat` is the ICF catalog name.

- `databaseName.objectName` is the name of the database and object containing the data set being recovered.

- `p` corresponds to the value that is used by DB2 for this node.

- `znnn` is the data set number.

`z` represents the letter A, B, C, D, or E. These letters correspond to the first digit—0, 1, 2, 3, or 4—of the partition number. `nnn` represents the rest of the partition number. `znnn` represents the partitions numbers as follows:

- A001 through A999 for partitions 1 through 999
- B000 through B999 for partitions 1000 through 1999
- C000 through C999 for partitions 2000 through 2999
- D000 through D999 for partitions 3000 through 3999
- E000 through E096 for partitions 4000 through 4096
When you want to specify a name of your own choosing, use the MODEL option to specify the name of the CLUSTER portion of the VSAM data set that you want to use. The data set name that you specify may include any of the symbolic variables that are listed in the description of the MODEL keyword in the following section.

When you use INDEP OUTSPACE, you must specify it after the object specification, as shown in the syntax diagram in Figure 11 on page 155. Also, see “Testing recovery with simulation” on page 366 and Chapter 5, “Examples of RECOVER PLUS jobs” for information about applications of this feature.

**MODEL**

The MODEL option specifies the non-DB2 data set cluster name to which you want to redirect the output from the recovery of the specified object. The data set name that you specify may optionally include any of the symbolic variables in Table 9:

<table>
<thead>
<tr>
<th>Symbolic variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;JOBNAME</td>
<td>JOB name used in the JCL</td>
</tr>
<tr>
<td>&amp;STEPNAME</td>
<td>STEP name used in the JCL (PROC names are ignored)</td>
</tr>
<tr>
<td>&amp;VCAT</td>
<td>VCAT name for the space</td>
</tr>
<tr>
<td>&amp;DB</td>
<td>database containing the space</td>
</tr>
<tr>
<td>&amp;TS</td>
<td>table space or index space</td>
</tr>
<tr>
<td>&amp;SP</td>
<td>table space or index space</td>
</tr>
<tr>
<td>&amp;DSNUM or &amp;PART</td>
<td>data set or partition (2 digits for 0 - 99, 3 digits for 100 - 999, 4 digits for 1000 – 4096)</td>
</tr>
<tr>
<td>&amp;LDSNUM or &amp;LPART</td>
<td>data set or partition (3 digits for 000–999 and 4 digits for 1000–4096)</td>
</tr>
<tr>
<td>&amp;USERID or &amp;UID</td>
<td>job or TSO user ID</td>
</tr>
<tr>
<td>&amp;SSID</td>
<td>DB2 subsystem ID</td>
</tr>
<tr>
<td>&amp;ATTACH</td>
<td>DB2 group attachment name or subsystem ID</td>
</tr>
<tr>
<td>&amp;DATE</td>
<td>current date (in the form YYMMDD)</td>
</tr>
<tr>
<td>&amp;TIME</td>
<td>current time (in the form HHMMSS)</td>
</tr>
<tr>
<td>&amp;JDATE</td>
<td>current Julian date (in the form YYDDDD)</td>
</tr>
<tr>
<td>&amp;YEAR</td>
<td>current year (in the form YY)</td>
</tr>
<tr>
<td>&amp;MONTH</td>
<td>current month (in the form MM)</td>
</tr>
<tr>
<td>&amp;DAY</td>
<td>current day (in the form DD)</td>
</tr>
<tr>
<td>&amp;JDAY</td>
<td>current Julian day (in the form DDD)</td>
</tr>
<tr>
<td>&amp;HOUR</td>
<td>current hour (in the form HH)</td>
</tr>
<tr>
<td>&amp;MINUTE</td>
<td>current minute (in the form MM)</td>
</tr>
<tr>
<td>&amp;SECOND</td>
<td>current second (in the form SS)</td>
</tr>
<tr>
<td>&amp;UTIL</td>
<td>utility ID (truncated to eight characters)</td>
</tr>
</tbody>
</table>
Point-in-time recovery specification

The syntax diagram for the Point-in-Time Recovery Specification is in Figure 16 on page 159.

**NOTE**
Place syntax for the Point-in-time recovery specification after the syntax for Table space specification (Figure 12 on page 156), Single index specification (Figure 13 on page 157), or Multiple index specification (Figure 14 on page 158).

**TORBA or TOLOGPOINT**

Use TORBA or TOLOGPOINT to recover a table space or index to a prior point in time identified by a log point. (For more information about using TORBA/TOLOGPOINT with table spaces and indexes, see “RECOVER INDEX and RECOVER INDEXSPACE command dependencies and prohibitions” on page 165.)

You can use TORBA or TOLOGPOINT interchangeably, regardless of the version of DB2 that you are using.

A TOLOGPOINT recovery returns a space to a point that reflects all activity on the space up to and including a log record at the log point specified. Any log record or activity higher than TOLOGPOINT is no longer reflected in the space but may be reinstated by a subsequent TOLOGPOINT recovery.

**NOTE**
If you are recovering directly to the real DB2 data sets and are not using INDEPENDENT OUTSPACE, a point-in-time recovery event is recorded in SYSIBM.SYSCOPY for table spaces and COPY YES indexes or BMCXCOPY for COPY NO indexes. However, if you are using the INDEPENDENT OUTSPACE option to recover to non-DB2 data sets and specify TORBA or TOLOGPOINT, the event is not recorded in SYSIBM.SYSCOPY or BMCXCOPY because the actual DB2 data sets are not changed.

Use of OUTCOPY ONLY with TORBA or TOLOGPOINT is not a point-in-time recovery. It is simply creation of a copy as of a prior point in time. The copy may be registered in SYSCOPY or BMCXCOPY, but no other event is recorded.

### Table 9  Symbolic variables used with INDEP OUTSPACE and their descriptions (part 2 of 2)

<table>
<thead>
<tr>
<th>Symbolic variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;SEQ</td>
<td>a sequential number that restarts at 1 at the beginning of each job step execution</td>
</tr>
<tr>
<td>&amp;INST</td>
<td>instance number, with valid values of 1 or 2</td>
</tr>
</tbody>
</table>
TORBA LASTQUIESCE (relativeGenerationNumber) or TOLOGPOINT LASTQUIESCE (relativeGenerationNumber)

Use TORBA LASTQUIESCE or TOLOGPOINT LASTQUIESCE to recover to the most recent quiesce point registered in SYSIBM.SYSCOPY for the table space being recovered. For index recovery, the most recent quiesce of the table space is used.

Optionally, you can add a relative generation number in parentheses to indicate the quiesce to use. A relative generation number of (0) uses the most recent image copy. For example, TOLOGPOINT LASTQUIESCE (-1) indicates the use of the quiesce before the most recent quiesce.

If you are recovering multiple table spaces in a single RECOVER TABLESPACE command statement, all of the table spaces must have a common last quiesce point. However, if you are using multiple RECOVER TABLESPACE command statements to recover multiple table spaces, each table space can have a different last quiesce point. The same is true for index recovery with RECOVER INDEX or RECOVER INDEXSPACE.

LASTQUIESCE can be abbreviated as LASTQ.

NOTE

If you use LASTQUIESCE and the quiesce is prior to an ALTER ADD PART, the recovery will fail. You must specify a hard-coded RBA. If the partition was added for a partition-by-growth universal table space, you can use the LASTQUIESCE option.

TORBA LASTCOMMONQ or TOLOGPOINT LASTCOMMONQ

Use TORBA LASTCOMMONQ or TOLOGPOINT LASTCOMMONQ to recover a set of table spaces and indexes to the most recent common quiesce point registered in SYSIBM.SYSCOPY. Index recoveries are based on the common quiesce point of the owning table space. If no common quiesce point exists for all table spaces and indexes specified, an error message is issued.

If you use this option when only one table space or index is recovered, the effect is exactly the same as using TORBA LASTQUIESCE.

You can use TORBA LASTCOMMONQ or TOLOGPOINT LASTCOMMONQ with RECOVER INDEX (ALL) or when multiple index space specifications exists in one command statement.

TORBA LASTARCHQ or TOLOGPOINT LASTARCHQ

Use TORBA LASTARCHQ or TOLOGPOINT LASTARCHQ to recover to the log point of the last ARCHIVE LOG MODE(QUIESCE) command issued for the DB2 subsystem.
When you use this option, RECOVER PLUS issues a message indicating the date and time of the ARCHIVE LOG MODE (QUIESCE) command used.

**TORBA LASTSHUTDOWN** or **TOLOGPOINT LASTSHUTDOWN**

Use **TORBA LASTSHUTDOWN** or **TOLOGPOINT LASTSHUTDOWN** to recover to the log point of the last normal system shutdown. When you use this option, RECOVER PLUS issues a message indicating the date and time of the last successful system shutdown. You cannot use the LASTSHUTDOWN keyword to recover to the log point of an abnormal DB2 termination. When you use this keyword, ensure that the table spaces and indexes were not in an exception status at the shutdown.

---

**NOTE**

In a data sharing environment, this keyword is used to recover to the last successful shutdown of the member subsystem on which the job is running, without regard to other member subsystems. For this reason, use extreme caution when you use **TORBA LASTSHUTDOWN** or **TOLOGPOINT LASTSHUTDOWN** in a data sharing environment to ensure that the shutdown used is really a point of data consistency.

To ensure that the last successful shutdown is really a point of data consistency, one of the following conditions must be true:

- The subsystem on which the RECOVER PLUS job is running was the last member of a data sharing group to be stopped.
- All updates to the table spaces and indexes to be recovered occurred on the subsystem on which the RECOVER PLUS job is running.

---

**TORBA LOGMARK logMarkName (logMarkGeneration) or TOLOGPOINT LOGMARK logMarkName (logMarkGeneration)**

Use **TORBA LOGMARK logMarkName** or **TOLOGPOINT LOGMARK logMarkName** to recover to a prior point in time identified by a log mark that was created in Log Master.

Optionally, you can add a log mark generation number in parentheses, *(logMarkGeneration)*. If you do not specify a log mark generation, RECOVER PLUS uses the most recent version of the log mark. You can use one of the following ways to specify the log mark generation:

- If you specify a log mark generation as less than or equal to zero (*\( \leq 0 \)*), RECOVER PLUS treats the generation as a relative generation. RECOVER PLUS refers to the most recent log mark with the generation of zero (*0*). The previous generation of the log mark is referred to with (*-1*), and so on.

- If you specify a log mark generation as greater than zero (*\( > 0 \)*), RECOVER PLUS treats the generation as an absolute generation number and uses the specified version of the log mark.
When you create the log mark, you can specify if Log Master should create a quiesce point. You can also specify that Log Master set a log mark at a quiet point or a non-quiet point. If you know that the log mark is at a quiesce or a quiet point for the space or spaces that you are recovering, use this option to perform a point-in-time recovery. If the log mark is not at a quiet point, you should use a timestamp recovery using OPTIONS RECOVERYPOINT LOGMARK logMarkName (page 143).

**NOTE**

You must determine if the log mark is at a quiesce or quiet point or not. If you are not sure that there is a quiesce or quiet point, a timestamp recovery may be best although it will take longer to look for inflight transactions.

**TORBA X'logPoint' or TOLOGPOINT X'logPoint'**

Use TORBA X'logPoint' or TOLOGPOINT X'logPoint' to recover to a prior point in time identified by the log point, 'logPoint'. Except with BACKOUT, only log records with starting log points less than or equal to 'logPoint' are used by RECOVER PLUS.

'logPoint' is a string of up to twelve hexadecimal digits.

**BACKOUT**

BACKOUT invokes the backout strategy for point-in-time recovery by using log points (TORBA, TOLOGPOINT). This strategy assumes that spaces are undamaged and that you require a reset to a specific point in time. The spaces are used with the log records between the point in time and the current point to back out to the required state. Using BACKOUT may enhance the performance of a point-in-time recovery significantly.

The following restrictions apply:

- All recovery requests in SYSIN must use BACKOUT. You cannot mix backward and forward recovery in SYSIN.

- The space must be current as of the last logged activity and not damaged in any way. Multi-data-set, nonpartitioned spaces must have all data sets recovered (DSNUM ALL).

- The space must not be in RECP, RECP*, RBDP, RBDP*, PSRCP, PSRBD, GRECP, WEPR, REFP, or STOPE status or have an LPL range.
No LOAD or REORG events can exist between the log point specified and the current point in time. For indexes, no REBUILD INDEX events can exist in this range.

No prior point-in-time recovery with a START_RBA greater than the log point requested and a PIT_RBA less than the log point requested can exist.

BACKOUT may not be requested with the following RECOVER PLUS options:

— INDEP OUTSPACE
— OUTCOPY ONLY
— OBIDXLAT
— DROPRECOVERY
— LOGONLY or LOGAPPLY ONLY

NOTE

BACKOUT uses only logs and spaces but requires that the spaces be current. The LOGONLY and LOGAPPLY ONLY options imply applying log records going forward by using a space restored to a previous state.

Change accumulation files are not allowed with BACKOUT because they are not properly ordered. Output accumulation files are also not supported because they are defined from the point of the last image copy to the current point in time.

TOCOPY specification

The syntax diagram for the TOCOPY Specification is in Figure 17 on page 159.

TOCOPY

The TOCOPY option specifies the last image copy to be processed by RECOVER PLUS to recover the specified table space (full or incremental copies) or index (full copies only). The recovery process ends after this image copy has been applied. If you specify an incremental copy for table space recovery, the prior full copy and any intervening incremental copies are also used in the restore process. Except when used with the INCOPY option, the image copy specified must be registered in SYSIBM.SYSCOPY (or in BMCXCOPY for indexes) as an image copy or as a “kept” incremental copy (as specified by the BMC COPY PLUS for DB2 product) for the specified object or object data set. When you specify TOCOPY, the log is not used to apply updates that are made after the copy.
After a TOCOPY recovery, the data in the space reflects the data as of the time of the specified image copy. Any log record or activity after the TOCOPY recovery is no longer reflected in the space but may be reinstated by a subsequent TOLOGPOINT recovery.

You can specify the image copy by its data set name or by the keyword LASTCOPY.

If the copy named was created by using SHRLEVEL CHANGE, RECOVER PLUS issues a warning (message BMC40971), but continues processing. (With a SHRLEVEL CHANGE copy, the pages may not be logically consistent.)

**NOTE**

When you use TOCOPY, you may specify only one table space. To recover more than one table space by using TOCOPY, you can use multiple RECOVER TABLESPACE command statements. The same is true for index recovery with RECOVER INDEX or RECOVER INDEXSPACE.

If you are recovering directly to the real DB2 data sets (not using INDEPENDENT OUTSPACE and not using OUTCOPY ONLY), a point-in-time recovery event is recorded in SYSIBM.SYSCOPY or BMCXCOPY. However, if you are not changing the actual objects and specify TOCOPY, the event is not recorded in SYSIBM.SYSCOPY (or BMCXCOPY) because the actual DB2 data sets are not changed.

**TOCOPY LASTCOPY (relativeGenerationNumber)**

TOCOPY LASTCOPY specifies that the most recent image copy (full or incremental for table spaces, full for indexes) registered in SYSIBM.SYSCOPY (or BMCXCOPY for indexes) is the last one to be processed by RECOVER PLUS to recover the specified object.

Optionally, you can add a relative generation number in parentheses to indicate the copy to use. A relative generation number of (0) uses the most recent image copy. For example, TOCOPY LASTCOPY (-1) indicates the use of the copy before the most recent copy.

When used with the INCOPY option, LASTCOPY means the last copy specified in your INCOPY list.

**TOCOPY dataSetName**

TOCOPY *dataSetName* specifies that the image copy data set named (full or incremental for table spaces, full for indexes) is the last one to be processed by RECOVER PLUS to recover the object. The named copy must be registered in SYSIBM.SYSCOPY (or BMCXCOPY for indexes or Instant Snapshot copies) unless you are choosing the INCOPY option. In that case the named copy must be the last copy specified in your INCOPY list. When the data set is part of a generation data group (GDG), you can specify either the full name or the relative GDG name.
If you specify a relative GDG name for an existing data set, use either (0) or (-n) for the relative generation. Do not refer to it as the (+1) generation. The most recent data set is the (0) generation, even if it was created in a previous step in the same job.

If the image copy is an Instant Snapshot copy, you should specify the VSAM data component name or the cluster component name. (The VSAM data component name is printed in the output from the COPY PLUS job that created the Instant Snapshot.) In reporting, RECOVER PLUS always shows only the data component name.

The following restrictions apply to the TOCOPY dataSetName option:

- You cannot specify a data set named LASTCOPY when you specify TOCOPY data set names.

- If dataSetName is not cataloged, you must use the TOVOLUME option to complete the identification. TOVOLUME identifies the volume containing the image copy data set to be used for recovery. You can use TOVOLUME only with TOCOPY dataSetName. The value that you specify with TOVOLUME depends on whether the data set is cataloged or uncataloged.

**TOVOLUME CATALOG**

TOVOLUME CATALOG is the default and identifies the image copy data set as a cataloged data set. It specifies that the volume serial number should be obtained from the operating system catalog.

**TOVOLUME volumeSerialNumber**

TOVOLUME volumeSerialNumber identifies the image copy data set as uncataloged, where volumeSerialNumber is the serial number of the first volume containing the image copy data set. (If the copy is in multiple volumes, the serial number of the first volume is the only one you need to specify. RECOVER PLUS determines all other volumes from SYSCOPY or BMCXCOPY.)

**TOSEQNO integer**

The integer parameter for TOSEQNO is the sequence number of the data set containing the image copy data set on the tape volume. You can use TOSEQNO only when you specify both TOCOPY and TOVOLUME volumeSerialNumber. If you specify TOCOPY and TOVOLUME volumeSerialNumber (to indicate an uncataloged data set) but do not specify TOSEQNO, RECOVER PLUS assumes a value of 1 if the copy is cataloged. Otherwise, RECOVER PLUS searches SYSCOPY or BMCXCOPY for a matching volume and uses the sequence number found.
LOGSORT specification

For a description of the LOGSORT Specification options, see “LOGSORT work data set specification description” on page 113. The syntax diagram for the LOGSORT Specification is in Figure 18 on page 159.

LOGAPPLY ONLY

LOGAPPLY ONLY specifies that only log information is to be used to update a table space or index and that no image copy data sets are to be used. Use this option when you want to specify a FROMRBA or FROMLOGPOINT.

LOGAPPLY ONLY is useful when another job has already recovered one or more table spaces or indexes to a known state. LOGAPPLY ONLY directs RECOVER PLUS to avoid merging and restoring copies and to apply only log updates from the FROMRBA or FROMLOGPOINT specified.

When you want to use only log records to update multiple table spaces or indexes in a single RECOVER TABLESPACE, RECOVER INDEX, or RECOVER INDEXSPACE command statement, you need to specify only a single LOGAPPLY ONLY. However, you must specify FROMRBA or FROMLOGPOINT for each table space or index.

When you use LOGAPPLY ONLY, you cannot use the TOCOPY option, the INCOPY option, or the INLOG option.

LOGONLY

LOGONLY applies only the log records to the data sets. All log records written after a point recorded in the data set are applied. Specify LOGONLY when the data sets of the target objects have been restored by using another process and the HPGRBRBA value in the header pages are correctly set to indicate where log apply should begin. You can use this option on single table spaces or indexes, or on table space or index lists.

When you use LOGONLY, you cannot use the TOCOPY option, the INCOPY option, or the INLOG option.

LOCALSITE

LOCALSITE indicates that only the image copies created for the local site are used in the recovery. The image copies are used in the order specified in the default options module for local site:

BMC961111I DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE
BMC961131 COPIES = (LP, LB) (DEFAULT SEQUENCE FOR IMAGE COPIES)
RECOVERYSITE

RECOVERYSITE indicates that only the image copies for the recovery site are used in the recovery. The image copies are used in the order specified in the default options module for remote site:

<table>
<thead>
<tr>
<th>BMC961121</th>
<th>DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC961131</td>
<td>COPIES = (RP, RB) (DEFAULT SEQUENCE FOR IMAGE COPIES)</td>
</tr>
</tbody>
</table>

CLONE

The CLONE option indicates that RECOVER PLUS is to recover clone table or index data. When you recover with clone data, the following limitations apply:

- You cannot refer to the same object with the CLONE option and without the CLONE option in the same recovery step; you cannot process the base and its clone in the same command.
- Related objects (table space and all indexes for its table, all related LOB objects) should use the same CLONE specification in the same recovery step.

ANALYZE

The ANALYZE option prints a recovery plan before executing that plan. The following information about objects is included in the plan:

- names of any image copy data sets on which activities are or will be based
- names of any log data sets on which activities are or will be based
- name of any change accumulation files on which activities are or will be based
- log ranges, if any, on which activities are or will be based
- number of log pages to be read
- record sizes for index sort work data sets
- phases to occur during execution
- steps to occur within each phase

Use this information to allocate space more accurately for log sort work data sets, index sort work data sets, and index key work data sets, thereby limiting abends that are caused by inadequate data set allocations. You can use these statistics with historical information about the times required to perform the various operations to estimate recovery time.
NOTE

If you coded ANALYZE with the OPTIONS command, you must code the same value here. If you code multiple RECOVER command statements you must use the same value for the ANALYZE option in all of the command statements. If you specify a value for ANALYZE on one command statement and take the default on the others, the value that you specify on the single command statement becomes the default for the others.

ANALYZE YES

ANALYZE YES is the default and provides the information listed in the preceding section.

ANALYZE NO

If you specify ANALYZE NO, RECOVER PLUS does not provide any recovery plan information, but does provide object summaries.

ANALYZE ONLY

If you specify ANALYZE ONLY, RECOVER PLUS provides the same information as it provides with ANALYZE YES and then stops the job when the ANALYZE phase is complete. You can use the data from the ANALYZE phase to determine the resources required and what will happen during the recovery run. You cannot restart a RECOVER PLUS job that specifies ANALYZE ONLY, but you can start a new recovery job.

Use the ANALYZE ONLY option to determine which phases will occur with a specific recovery request, or which copy data sets and log data sets will be used in the recovery.

If you provide a SYSPICK DD statement, RECOVER PLUS generates a list of all of the input tape and cartridge volumes that are allocated during recovery. For more information about SYSPICK, see “RECOVER PLUS data sets and RECOVER PLUS DD statements” on page 276.

If you specify OPTIONS EARLYRECALL with ANALYZE ONLY, the recall of the data sets needed for the recovery will be initiated. You must code OPTIONS EARLYRECALL explicitly to initiate the recalls, because OPTIONS NOEARLYRECALL is the default for ANALYZE ONLY executions.
REDEFINE

Use the REDEFINE option to avoid the normal deletion and reallocation of the object data sets for STOGROUP-defined objects. This option is ignored for VCAT-defined object data sets. The value that you specify for REDEFINE applies to all STOGROUP-defined objects included with a single RECOVER command. To recover multiple STOGROUP-defined objects with some reallocated and others not reallocated, you must use one RECOVER command statement for the objects to be reallocated and a second RECOVER command statement for those objects not to be reallocated.

Regardless of the value of REDEFINE, if the data sets for a STOGROUP-defined object do not exist, RECOVER PLUS creates the VSAM data sets.

NOTE

When a point-in-time recovery of a multi-data-set, nonpartitioned object is performed, more or fewer data sets may result than were previously used. If data sets were added after the point-in-time recovery point, there are fewer data sets after the point-in-time recovery. If the object was reorganized after the point-in-time recovery point, there may be more data sets.

REDEFINE YES

REDEFINE YES is the default. If you specify REDEFINE YES, RECOVER PLUS deletes and reallocates the VSAM data sets for STOGROUP-defined objects prior to the recovery. For more information, see “Storage group-defined data sets” on page 92.

NOSCRATCH

If you specify NOSCRATCH with REDEFINE YES, RECOVER PLUS invokes IDCAMS to uncatalog without scratching. You can use NOSCRATCH to support disaster recovery scenarios where the following conditions are true:

- DASD volumes at the recovery site differ from those at the primary site.
- The operating system catalog and the DB2 catalog and directory are restored before running RECOVER PLUS.
- DB2 STOGROUPS are changed to point to the recovery site packs.

Using this option eliminates mount messages for the primary site packs that do not exist at the recovery site.
RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE option descriptions

**NOTE**
REDEFINE YES has no effect if you specify INDEP OUTSPACE. For the instructions for creating data sets for INDEP OUTSPACE, see “INDEPENDENT OUTSPACE specification” on page 178.

If an Instant Snapshot copy is used to recover the object, REDEFINE YES has no effect and the data sets are processed as if REDEFINE NO had been specified.

**REDEFINE NO**

REDEFINE NO tells RECOVER PLUS not to delete and reallocate the VSAM data sets for the object. This option enables you to treat STOGROUP-defined objects as VCAT-defined objects and to reallocate the VSAM data sets manually, or use the data sets already assigned to the object. New volumes are used as needed from the STOGROUP, even with REDEFINE NO specified.

**NOTE**
With REDEFINE NO specified, when recovery of a multi-data-set, nonpartitioned object is performed as DSNUM ALL, any necessary additional data sets are created and unused data sets are deleted.

When you specify REDEFINE NO, RECOVER PLUS does not free unused extents in a retained data set. If you want the recovery to free unused extents, specify REDEFINE YES, which is the default.

**REUSE**

The REUSE option specifies that the space will not be deleted or redefined. This option cannot be used with the REDEFINE option.

**UPDATE VERSIONS**

Use the UPDATE VERSIONS option to have RECOVER PLUS call the DB2 REPAIR VERSIONS utility after a table space recovery of a versioned table space is complete. The DB2 REPAIR VERSIONS utility updates the version information in the DB2 catalog and directory so that the source and target information match.

The UPDATE VERSIONS option is useful if you migrate data from one DB2 system to another using RECOVER PLUS. Using RECOVER PLUS is a fast way to restore a copy from the source system to the target system.

If you use RECOVER PLUS and this option to perform migration, you need to

- make sure that the oldest version in your copy is greater than 0
- recover table spaces in a separate step from rebuilding indexes
OBIDXLAT specification

The syntax diagram for the OBIDXLAT specification is in Figure 19 on page 160. The following rules apply to the OBIDXLAT specification:

- The source or the target DBIDs and PSIDs may be omitted, or the DBID or PSID keyword may be omitted. If a source or target value is omitted from a DBID or PSID clause, the missing value defaults to the DBID or PSID of the object as reflected in the catalog of the current subsystem except when INCOPY or OBJECTSET is specified without DROPRECOVERY. If INCOPY or OBJECTSET is specified without DROPRECOVERY, the source DBID and PSID is taken from the image copy.

- For a single-table table space, the source or the target OBID may be omitted, or the OBID keyword may be omitted. If a source or target value is omitted from an OBID clause, the missing value defaults to the OBID of the object as reflected in the catalog of the current subsystem except when INCOPY or OBJECTSET is specified without DROPRECOVERY. If INCOPY or OBJECTSET is specified without DROPRECOVERY, the source OBID is taken from the image copy. If more than one OBID clause is coded, the source and target values are required.

- For multiple-table table spaces, rows are not translated for which there is no OBID clause, but you must code at least one OBID clause with both source and target values, or you must code a DBID or PSID clause.

- For an index, if you code an OBID clause, you must code two OBID clauses, one for the index and one for the table on which the index is built. When INCOPY or OBJECTSET is specified for an index, you may omit both OBID clauses, in which case the source OBID for the index and the table is taken from the image copy. If you do not omit the ID clauses, you must code both the target and source values for both OBID clauses.

If you code INCOPY and OBIDXLAT, you must code TOCOPY or you must supply the source DBID and PSID in the clause. You can use the USEHDROBIDS option (page 123 and page 557) to indicate if the OBIDs in the header are valid or not.

If you code OBIDXLAT and log apply is required, you should also specify OUTCOPY ONLY. These specifications produce an output copy that you can use on a different system with different OBIDs. If you were to do OBIDXLAT to an existing space, the object IDs would no longer match.

For example, these are valid specifications:

| RECOVER ... OBIDXLAT DBID(100,101) PSID (200,201) |
| INCOPY FULL DSNAME ... RBA X'123456' |

or

Chapter 3  RECOVER PLUS syntax  193
RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE option descriptions

The following example is not valid:

```
RECOVER ... OBIDXLAT DBID('101') PSID('201')
INCOPY FULL DSNAME ...
TOCOPY LASTCOPY
```

**OBIDXLAT**

Use the OBIDXLAT keyword to change the internal IDs in the DB2 table space or index as the table space or index is recovered.

**NOTE**

Note the following information about the use of OBIDXLAT:

- When you use Instant Snapshots for migration and specify the OBIDXLAT option, RECOVER PLUS automatically detects if the OBIDs have not changed and avoids the overhead of further processing.

- OBIDXLAT of Instant Snapshots of universal table spaces (UTS) further reduces the processing time on DB2 Version 9, even if OBIDs change.

- If you specify RESET, RECOVER PLUS does not get a speed improvement. For data sharing environments, RESET is usually not necessary. For non-data-sharing environments, if the RBAs on the target system are larger than the RBAs on the source, RESET is probably not necessary.

**RESET**

The RESET keyword causes the log points in each data page to be reset to '0'. You must specify RESET when you are migrating data from one DB2 non-data-sharing system to another because the log point values on the first subsystem is meaningless or misleading on the target DB2.

The recovery log point and level fields in the header page are also reset to 0 if you are not using the INDEPENDENT OUTSPACE option. If you are using the INDEPENDENT OUTSPACE option, the level fields are reset but the recovery log point is not.

```
DBID ('hexSourceID', 'hexTargetID')
DBID (decimalSourceID, decimalTargetID)
```

Optionally, specify this clause to provide the DBIDs to be translated. You can specify the DBID in hexadecimal or decimal format. For more information, see “OBIDXLAT specification” on page 193.
RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE option descriptions

PSID (X'hexSourceID', X'hexTargetID')
PSID (decimalSourceID, decimalTargetID)

Optionally, specify this clause to provide the PSIDs to be translated. You can specify the PSID in hexadecimal or decimal format. For more information, see "OBIDXLAT specification" on page 193.

When you are recovering clone objects, remember that the base and clone objects are differentiated by differences in the PSID value. The high order bit of the PSID is used to refer to a particular data set instance number as follows:

- a high order bit value of 0 indicates instance number 1
- a high order bit value of 1 indicates instance number 2

Other than the high order bit, the PSID numbers are the same for both a base object and its clone. You should use the PSID with the high order bit set on for instance 2 objects in the OBIDXLAT translation specification.

OBID (X'hexSourceID', X'hexTargetID')
OBID (decimalSourceID, decimalTargetID)

Optionally, specify this clause to provide the OBIDs to be translated. You can specify the OBID in hexadecimal or decimal format. For more information, see "OBIDXLAT specification" on page 193.

Non-registered copy or INLOG specification

The syntax diagram for the Non-Registered Copy or INLOG Use Specification is in Figure 20 on page 160.

INCOPY FULL

Specify the INCOPY clause to recover by using non-registered image copies. You must specify a full image copy. You may specify any number of incremental copies.

**NOTE**

When you specify INCOPY, RECOVER PLUS analyzes relevant events in SYSCOPY and BMCXCOPY, unless you specify TOCOPY LASTCOPY (see page 186). Then, no access to SYSCOPY or BMCXCOPY occurs.

INCOPY is recommended as the technique to specify a copy of an index made with DSN1COPY for use in an index recovery. Specification of the LOGPOINT and SHRLEVEL values are optional. If the index is encrypted, the ENCRYPTED option (page 206) is required.
If this recovery is not an INDEPENDENT OUTSPACE recovery or an OUTCOPY ONLY recovery, RECOVER PLUS will register a point-in-time recovery with a PIT_RBA of zeros. Further, unless registered output copies are being created, COPY PENDING is set on any table space so recovered.

**NOTE**

Note the following information about the use of INCOPY:

- INCOPY does not support the use of a cabinet copy created by the BMC COPY PLUS product. If you have only a cabinet copy and need to do a drop recovery, you can use the COPY PLUS COPY IMAGECOPY command to unstack a cabinet copy and register the copies in SYSCOPY. You can then use the individual copies with the INCOPY option.

- RECOVER PLUS supports Online Consistent Copies and consistent FlashCopies with log apply when you *do not* specify INCOPY.

- RECOVER PLUS supports Online Consistent Copies and consistent FlashCopies with INCOPY TOCOPY specified.

- RECOVER PLUS does *not* support Online Consistent Copies or consistent FlashCopies when you specify *both* INCOPY and log apply (RBA or LOGPOINT specified in the INCOPY specification).

**INLINE**

INLINE specifies that the copy specified in the INCOPY specification is an inline copy created by COPYDDN or RECOVERYDDN from a DB2 REORG or LOAD utility. To use an inline copy, you must specify this parameter. This option is valid for full image copies only. The INLINE option is not valid with indexes.

**SNAPSHOT**

SNAPSHOT specifies that the copy specified in the INCOPY specification is an Instant Snapshot copy that is not registered in BMCXCOPY. This option is valid for full image copies only. (For more information, see “Using Instant Snapshots” on page 429.)

**FLASHCOPY**

FLASHCOPY specifies that the copy specified in the INCOPY specification is an IBM FlashCopy that is registered in SYSCOPY. This option is valid for full image copies only.

**INCR**

INCR specifies that the copy is an incremental copy. The INCR copies are processed in the order in which you specified them. Specification of the RBA/LOGPOINT and SHRLEVEL values (see “INCOPY specification” on page 204) are optional.
NOCOPYPEND

If you specify the NOCOPYPEND option, RECOVER PLUS resets COPY-pending status and issues message BMC96232 to inform you that COPY-pending status has been reset even though the space is not recoverable.

For a point-in-time recovery, if you do not specify the NOCOPYPEND option, RECOVER PLUS sets COPY-pending status when all of the following conditions exist:

- You did not specify INDEP OUTSPACE.
- You did not specify OUTCOPY ONLY.
- There is no registered OUTCOPY.

You may want to use the NOCOPYPEND option if you are migrating data from one DB2 subsystem to another in a query or testing system where you do not need to be able to run a recovery on the target space. To migrate the data, you make an image copy on the source system, transport the copy to the target system if necessary, and use RECOVER INCOPY on the target system.

Because the input copy is not registered on the target system, the target space is not recoverable after the RECOVER INCOPY. You cannot run a RECOVER to current on the target system because no usable copies are registered in SYSCOPY or BMCXCOPY.

INLOG RBA X'logPoint' or INLOG LOGPOINT X'logPoint'

INLOG RBA or INLOG LOGPOINT is used when you want to perform recovery by using only log after a LOAD LOG YES or a REORG LOG YES where the event is no longer registered in SYSCOPY. To do so, you must be aware of a point in the log corresponding to the specific event. You must specify a log point that exactly corresponds to such a point; otherwise, the recover will not be able to apply the log and a severe internal error will result with completion code 12.

INLOG RBA or INLOG LOGPOINT specifies the log point in the log where application of the log records should begin for the object being recovered.

'logPoint' is a string of up to twelve hexadecimal digits.

When you use INLOG RBA or INLOG LOGPOINT, you cannot use the TOCOPY option. You also may not specify LOGONLY or LOGAPPLY ONLY. Also, you cannot specify INCOPY when you use INLOG RBA or INLOG LOGPOINT, and any output copies made with the OUTCOPY ONLY facility cannot be registered unless DROPRECOVERY is also specified.
You can use the INLOG RBA and INLOG LOGPOINT keywords interchangeably, regardless of the version of DB2 used.

**NOTE**

INLOG RBA and INLOG LOGPOINT are not valid for index recovery because index records are not logged completely for a LOG YES utility.

**FROMRBA or FROMLOGPOINT specification**

The syntax diagram for the FROMRBA or FROMLOGPOINT Specification is in Figure 21 on page 161.

You can specify FROMRBA or FROMLOGPOINT only with LOGAPPLY ONLY. When you use LOGAPPLY ONLY, you must specify FROMRBA or FROMLOGPOINT for each table space and index specification.

FROMRBA or FROMLOGPOINT specifies the log point in the log where application of the log records should begin for the object or object data set being recovered. You can specify the log point with a keyword or with a hexadecimal string.

This option assumes that the object or objects have been brought to an appropriate point by some other activity prior to this step.

You can use the FROMRBA and FROMLOGPOINT keywords interchangeably, regardless of the version of DB2 that you are using.

**FROMRBA LASTQUIESCE or FROMLOGPOINT LASTQUIESCE**

FROMRBA LASTQUIESCE or FROMLOGPOINT LASTQUIESCE specifies that the log point where application of the log records will begin for this object (or object data set) is the log point of the most recent quiesce point registered in SYSIBM.SYSCOPY. If this is recovery of an index, the QUIESCE entry for the table space is used. You can abbreviate LASTQUIESCE to LASTQ.

**NOTE**

If you use LASTQUIESCE and the quiesce is prior to the ALTER ADD PART, the recovery will fail. You must specify a hard-coded RBA. If the partition was added for a partition-by-growth universal table space, you can use the LASTQUIESCE option.

**FROMRBA LASTCOPY or FROMLOGPOINT LASTCOPY**

FROMRBA LASTCOPY or FROMLOGPOINT LASTCOPY specifies that the log point where application of the log records will begin for this object (or object data set) is the log point of the most recent image copy (full or incremental) registered.
FROMRBA LASTARCHQ or FROMLOGPOINT LASTARCHQ

FROMRBA LASTARCHQ or FROMLOGPOINT LASTARCHQ specifies that the log point of the most recent ARCHIVE LOG MODE(QUIESCE) command is the point where application of the log records will begin for this object (or object data set).

FROMRBA LASTSHUTDOWN or FROMLOGPOINT LASTSHUTDOWN

FROMRBA LASTSHUTDOWN or FROMLOGPOINT LASTSHUTDOWN specifies that the log point of the most recent successful STOP DB2 command is the point where application of the log records will begin for this object (or object data set).

NOTE

If you specify this keyword in a data sharing environment, log records since the last successful shutdown of the member subsystem on which the job is running are used for recovery, without regard to other member subsystems. For this reason, use extreme caution when you use FROMRBA LASTSHUTDOWN or FROMLOGPOINT LASTSHUTDOWN in a data sharing environment to ensure that the shutdown used is really a point consistent with the state of the spaces.

FROMRBA X'logPoint' or FROMLOGPOINT X'logPoint'

FROMRBA X'logPoint' or FROMLOGPOINT X'logPoint' specifies the log point in the log where application of the log records for this object will begin.

'logPoint' is a string of up to twelve hexadecimal digits.

OUTCOPY specification

The syntax diagram for the OUTCOPY Specification is in Figure 22 on page 161. For option descriptions for the OUTCOPYDDN and RECOVERYDDN specifications, which are included in the OUTCOPY Specification, see “OUTCOPYDDN specification” on page 208 and “RECOVERYDDN specification” on page 209.

OUTCOPY

You can specify OUTCOPY to make copies of the object or object data set being recovered. For each object or data set that you process, you can request up to four image copies. The default is no copies. If you are recovering to the real DB2 data sets and you request image copies, you can choose which copies, if any, to register in the SYSIBM.SYSCOPY table (or BMCXCOPY table for indexes). However, if you are using the INDEPENDENT OUTSPACE option to recover to non-DB2 data sets and you request image copies, you cannot register those copies.

Output copies are produced in the new SYSTEMPAGES YES format when possible.
When you use the OUTCOPY ONLY facility, you can register the copies if you do not use OBIDXLAT or INCOPY or specify DROPRECOVERY.

If the output copies that you want to make are allocated in the JCL, the OUTCOPYDDN and RECOVERYDDN options specify the DD names for those data sets. If the output copies are dynamically allocated, the OUTCOPYDDN and RECOVERYDDN options refer to the appropriate OUTPUT descriptor names. For more information, see “OUTCOPYDDN specification” on page 208 and “RECOVERYDDN specification” on page 209. The copies that you want to register are specified by the REGISTER option (page 202).

When you create an OUTCOPY and want to force the copy to the cylinder-managed space of an extended address volume (EAV), specify the SPACE keyword (page 204).

How the copies are made depends on whether the object is partitioned or nonpartitioned.

- If the object is partitioned, and OUTCOPY BYPART is specified or a specific DSNUM is coded in the object specification, a separate output copy is made for each partition. If you specified OUTCOPY ASCODED and DSNUM ALL with the RECOVER command, a single output copy is made for the object as a whole.

For more information about using OPTIONS OUTCOPY BYPART, see “OUTCOPY” on page 121.

- If the object is nonpartitioned and you are recovering multiple data sets by specifying DSNUM ALL, each output copy is for all of the data sets. If you are recovering multiple data sets by specifying each one individually with DSNUM integer, a separate output copy is made for each data set.

**OUTCOPY NO**

OUTCOPY NO is the default and specifies that no output copies are required.

**OUTCOPY YES**

OUTCOPY YES specifies that you want image copies or DSN1COPY-type copies of the table space or index (or data set) being recovered. You can make up to four image copies or up to four DSN1COPY-type copies. A DD statement or an OUTPUT command statement must exist for each copy. The ddnames for the copy DD statements are specified by the OUTCOPYDDN and RECOVERYDDN options.

**NOTE**

You cannot use the OUTCOPY YES option to produce Instant Snapshot copies.
OUTCOPY ONLY

OUTCOPY ONLY does not affect the table space, index, or data set named in the RECOVER command statement and enables you to build copies without stopping the object for access and without actually reading the space, thereby avoiding DASD contention. You can use this option to build copies for a shadow version of the object, to build a migration image, or to build a registered copy for the named object’s recovery.

**NOTE**

If you use OUTCOPY ONLY to register a copy of a space at a point in time that is prior to existing incremental copies, the full copy will have a later TIMESTAMP, but a lower START_RBA than the associated incremental copies. Subsequent use of the IBM MODIFY utility to do an age-based DELETE from SYSCOPY may delete some of the incremental copies without deleting the associated full copy that was created by RECOVER PLUS. The resulting gap in the incremental copies can cause errors in a subsequent recovery that uses the remaining incremental copies. BMC Software recommends using the COPY PLUS MODIFY command to ensure that a consistent set of recovery resources is maintained in this situation.

The following restrictions apply to the OUTCOPY ONLY option.

- You cannot use inline copies as input to OUTCOPY ONLY because RECOVER PLUS requires use of the object itself as a staging area to order the copies.

- You cannot use the OUTCOPY ONLY option with Instant Snapshot copies.

- You cannot use the OUTCOPY ONLY option with the point-in-time recovery option BACKOUT.

- You cannot register an OUTCOPY ONLY copy to the same log point as an existing full copy for the same space or index. OUTCOPY ONLY copies cannot be registered if INCOPY, INLOG, or OBIDXLAT is specified except when DROPRECOVERY is also specified.

- You cannot generate a cabinet copy.

You can combine this option with an R+/CHANGE ACCUM product ACCUM command statement involving the same object, but only if the recovery does not include TORBA, TOLOGPOINT, or TOCOPY options. For more information about the ACCUM command statement, see the R+/CHANGE ACCUM for DB2 User Guide.
If copies are registered, they are registered as SHRLEVEL REFERENCE if any one of the following options is specified:

- TORBA or TOLOGPOINT to a quiesce point of the table space
- TORBA or TOLOGPOINT to a system wide quiesce point resulting from an ARCHIVE LOG MODE(QUIESCE) command
- TOCOPY to a SHRLEVEL REFERENCE incremental copy
- TORBA or TOLOGPOINT to a system shut down in a non-data-sharing environment

For all other cases, copies are registered as SHRLEVEL CHANGE.

REGISTER

Use this option to indicate how many, if any, copies you want to register in the SYSIBM.SYSCOPY table. Index copies are registered in the BMCXCOPY table.

You can make and register up to four copies, all with the same log point and the same SYSIBM.SYSCOPY time stamp. The first copy specified by ddname with the OUTCOPYDDN option is registered as the local site primary copy. Similarly, the second copy is registered as the local site backup copy. Third and fourth copies specified with RECOVERYDDN are registered as recovery site primary and backup copies.

NOTE

You cannot make a backup copy without a primary copy. You can, however, make and register a recovery site copy without making a local site copy.

REGISTER ALL

REGISTER ALL is the default and specifies that you want to register all of the copies that RECOVER PLUS makes for the object in the SYSIBM.SYSCOPY or BMCXCOPY table.

REGISTER NONE

If you specify OUTCOPY YES REGISTER NONE, RECOVER PLUS makes up to four DSN1COPY-type copies.
REGISTER \((DDNameList)\)

OUTCOPY YES REGISTER \((DDNameList)\) specifies by ddname those image copies that you want to register in the SYSIBM.SYSCOPI or BMCXCOPY table. The ddname list is optionally enclosed in parentheses and that the ddnames must be separated by commas. For each name in the list, there must be a ddname, ddname prefix, or output descriptor specified in OUTCOPYDDN/RECOVERYDDN explicitly or implicitly as the default value.

**NOTE**
The entire OUTCOPYDDN/RECOVERYDDN specification can be omitted and RECOVER PLUS will use the default ddnames. Also note, REGISTER allows the use of any form of the defaults (BMCCPY or BMCCY, BMCCPZ or BMCCZ, and so on) to refer to the default ddnames. Both forms have the same meaning for the REGISTER specification if the corresponding ddname in OUTCOPYDDN/RECOVERYDDN is defaulted. For more information, see “OUTCOPYDDN specification” on page 208, “RECOVERYDDN specification” on page 209, and “Copy data set ddname construction for JCL-allocated data sets” on page 210.

RESETRTS

Use RESETRTS to have RECOVER PLUS reset the copy DB2 real-time statistics (RTS) if the copy is registered when you specify OUTCOPY YES or OUTCOPY ONLY.

**WARNING**
You should specify RESETRTS only if the OUTCOPY you are making is a copy that approximately represents the current state of the space. If you specify RESETRTS for a copy corresponding to a different point in time, the RTS may give misleading information about the time of the last copy and the changes made since the copy.

Several examples of how you might use RESETRTS follow:

- You might use RESETRTS if you want to make a copy of your system at a specified point in time, such as at midnight. Because making GROUP YES SHRLEVEL REFERENCE copies of the whole system is not usually practical, another approach is to issue an ARCHIVE LOG MODE(QUIESCE) command to get a system-wide point of consistency. You can then use RECOVER PLUS OUTCOPY to make copies of all of the spaces at the point of consistency. While RECOVER PLUS makes copies with OUTCOPY, the system is available for update. Using this method, you get consistent copies of the whole system at the desired point in time with minimal outage. This approach is sometimes referred to as the stealth copy procedure. You should know when you use this procedure that the copy is very close to the current state of the system, and you should use RESETRTS to reset the RTS copy statistics.
You might want to use RTS to trigger utility runs. For example, you might want to make a copy if RTS shows that the last copy was made more than 2 days ago. When you use RECOVER PLUS to recover a space and make an OUTCOPY, RECOVER PLUS resets the RTS copy statistics and updates the RTS timestamp for the last copy. However, if you run RECOVER PLUS OUTCOPY ONLY and do not specify RESETRTS, RECOVER PLUS does not reset RTS because the space was not updated and the output copy does not necessarily match the space. For example, the current date and time might be 10 AM on October 15, but the OUTCOPY might be for 10 PM on October 12.

**SPACE (primary, secondary) allocation unit**

Specify SPACE to set the output allocation parameters for the named descriptor in OUTCOPYDDN. This use of SPACE overrides the SPACE specification in the corresponding OUTPUT descriptor. Specify SPACE TRK to allocate the output in tracks. Specify SPACE CYL to allocate the output in cylinders. The default value for this option is CYL. Specify the values in parentheses, as in the following example: SPACE (200,100) CYL. If no SPACE specification is provided, the default is (5,5) CYL.

**INCOPY specification**

The syntax diagram for the INCOPY Specification is in Figure 23 on page 162.

**DSNAME dataSetName**

DSNAME dataSetName is the data set name of the copy used during recover.

For an Instant Snapshot copy or an IBM FlashCopy, this name should be the VSAM data component name or the cluster component name. The VSAM data component name is printed in the output from the COPY PLUS job that created the Instant Snapshot. For a FlashCopy, the cluster component name is printed in the output from the DB2 COPY utility. In reporting, RECOVER PLUS displays the name that was supplied with this parameter.

If you specify a generation data group (GDG) name, you can either specify the relative GDG name or the fully qualified data set name.

If you specify a relative GDG name for an existing data set, use either (0) or (-n) for the relative generation. Do not refer to it as the (+1) generation. The most recent data set is the (0) generation, even if it was created in a previous step in the same job.
MODEL *dataSetName*

MODEL *dataSetName* specifies the VSAM cluster created by an Instant Snapshot or Online Consistent Copy. *dataSetName* must include a node for the data set number represented by A&LDSNUM. This node can appear anywhere within *dataSetName*.

---

**NOTE**

When you use INCOPY with TRANSFORM, specify the VSAM data component instead of the VSAM cluster.

---

**DSSIZE (integerG)**

DSSIZE is used with TRANSFORM to specify the data set size of the source table space. In most cases, the default value should be sufficient.

---

**NOTE**

DSSIZE and TRANSFORM are part of the High-speed Structure Change process.

---

Valid values are 0, 1G, 2G, 4G, 8G, 16G, 32G, and 64G. If the table space is not partitioned, the default is 2G. If the table space is partitioned and not a LARGE table space (has a 4-byte RID), the DSSIZE is calculated based on the number of partitions. If the table space is a LARGE type, the data set size is read from the header page. If the DSSIZE is not stored in the header page, you must specify DSSIZE in the SYSIN.

See the *Recovery Management for DB2 User Guide* for more information.

**PIECESIZE (…)**

Use PIECESIZE with TRANSFORM to specify the data set size of the source nonpartitioning index space. In most cases, the default value should be sufficient.

---

**NOTE**

PIECESIZE and TRANSFORM are part of the High-speed Structure Change process.

---

Valid values are any integer that is a power of 2 between 256K through 64G. The PIECESIZE is calculated based on the value of ptshift, which is found in the header page. If ptshift is not valued (an index migrated from an early DB2 release), you must specify PIECESIZE.
See the *Recovery Management for DB2 User Guide* for more information.

**RBA X'logPoint' or LOGPOINT X'logPoint'**

RBA X'logPoint' or LOGPOINT X'logPoint' indicates the log point of the copy. The keywords RBA and LOGPOINT are interchangeable, regardless of the version of DB2 used.

To recover to the current point or to a quiesce point, you must code a log point value on the last copy specified so that RECOVER PLUS knows at what point in the log to start obtaining log records for the table space or index.

If no log point is specified on the last copy, you must recover TOCOPY and no log is processed.

**SHRLEVEL REFERENCE**

SHRLEVEL REFERENCE indicates that the share level of the copy is REFERENCE. If you do not specify a value for SHRLEVEL, the default is REFERENCE.

**SHRLEVEL CHANGE**

SHRLEVEL CHANGE indicates that the share level of the copy is CHANGE.

**ENCRYPTED**

Use the ENCRYPTED option to specify that the specified copy is an encrypted image copy made by COPY PLUS. For more information about using encrypted copies, see “Recovering encrypted copies” on page 462.

---

**NOTE**

Encryption is a feature of the Recovery Management for DB2 solution and requires a valid Recovery Management solution password.

**TIMESTAMP** *timestamp*

The TIMESTAMP option specifies the timestamp when the encrypted copy was registered in BMCXCOPY. The *timestamp* value determines which key from the key data set (specified by the KEYDSNAM installation option described on page 552) RECOVER PLUS uses to decode the copy. The *timestamp* value uses either of the following formats:

```
yyyy-mm-dd-hh.mm.ss
tyyyy-mm-dd-hh.mm.ss.nnnnnn
```
You can omit leading zeros from the month, day, or hour parts of the timestamp; you can omit trailing zeros from the microseconds part of the timestamp.

**COMPRESSED**

Use the COMPRESSED option to indicate that the specified image copy is a compressed copy of a compressed index made by DSN1COPY or COPY PLUS. For more information about COPY PLUS compressed copies of compressed indexes, see the COPY PLUS for DB2 Reference Manual section “Copying compressed indexes.”

**INVOLUME**

The INVOLUME specification gives the volume information for the copy indicated in the INCOPY specification.

**INVOLUME CATALOG**

INVOLUME CATALOG is the default and identifies the image copy data set as a cataloged data set. It specifies that the volume serial number should be obtained from the operating system catalog.

**INVOLUME volSerialNumberList**

INVOLUME volSerialNumberList identifies the image copy data set as uncataloged where volSerialNumberList provides the serial numbers of the volumes containing the image copy data set. Include all volume serials in the order needed. volSerialNumberList specifies a list of one or more volume serial numbers, separated by commas.

**INDEVT deviceType**

The deviceType parameter for INDEVT identifies the device on which the copy resides. It is required for uncataloged data sets.

**INSEQNO integer**

The integer parameter for INSEQNO is the sequence number of the data set containing the image copy data set on the tape volumes. If you specify INCOPY and INVOLUME volumeSerialNumber (to indicate an uncataloged data set) but do not specify INSEQNO, RECOVER PLUS assumes file sequence number one.
OUTCOPYDDN specification

The syntax diagram for the OUTCOPYDDN Specification is in Figure 24 on page 162.

OUTCOPYDDN (DDName1,DDName2)

You can use OUTCOPYDDN (DDName1,DDName2) to override the default ddnames or ddname prefixes that are used by RECOVER PLUS for a table space or index to specify copy data set ddnames in the JCL. OUTCOPYDDN can also reference an output descriptor named with an OUTPUT command statement if the copy data set is to be dynamically allocated.

NOTE

The following restrictions apply only if OUTCOPYDDN refers to a DD statement in the JCL:

- When a table space or index has fewer than 100 partitions, you can specify any value of up to six characters. If copies are being made by partition, this value is a prefix. If DSNUM ALL and OPTIONS OUTCOPY ASCODED are specified, this value is a ddname.

- For a table space or index with 100 or more partitions, the default values for all partitions are BMCCY and BMCCZ, and you can specify any value of no more than five characters.

- For a nonpartitioned table space or index, if DSNUM ALL is specified or the highest value of DSNUM in the run is less than 100, you can specify any value up to six characters. If the highest value of DSNUM is 100 or higher, you can specify any value of up to five characters.

The ddnames specified with OUTCOPYDDN are optionally enclosed in parentheses but must be separated by a comma. (For more information, see “Copy data set ddname construction for JCL-allocated data sets” on page 210.)

OUTCOPYDSN (dataSetName1, dataSetName2)

You can use OUTCOPYDSN (dataSetName1, dataSetName2) to override the data set name specified by an output descriptor for OUTCOPYDDN for a dynamically allocated output image copy. Use of OUTCOPYDSN when the output image copy is not dynamically allocated is an error.
RECOVERYDDN specification

The syntax diagram for the RECOVERYDDN Specification is in Figure 25 on page 163.

RECOVERYDDN (DDName3,DDName4)

You can use RECOVERYDDN (DDName3,DDName4) to override the default ddnames or ddname prefixes that are used by RECOVER PLUS for a table space or index to specify copy data set ddnames in the JCL. RECOVERYDDN can also reference an output descriptor named with an OUTPUT command statement if the copy data set is to be dynamically allocated.

The following restrictions apply only if RECOVERYDDN refers to a DD statement in the JCL:

- When a table space or index has fewer than 100 partitions, you can specify any value of up to 6 characters. If copies are being made by partition, this value is a prefix. If DSNUM ALL and OPTIONS OUTCOPY ASCODED are specified, this value is a ddname.

- For a table space or index with 100 or more partitions, the default values for all partitions are BMCRY and BMCRZ, and you can specify any value of no more than 5 characters.

- For a nonpartitioned table space or index, if DSNUM ALL is specified or the highest value of DSNUM in the run is less than 100, you can specify any value up to six characters. If the highest value of DSNUM is 100 or higher, you can specify any value of up to five characters.

The ddnames specified with RECOVERYDDN are optionally enclosed in parentheses but must be separated by a comma. (For more information, see “Copy data set ddname construction for JCL-allocated data sets.”)

RECOVERYDSN (dataSetName3, dataSetName4)

You can use RECOVERYDSN (dataSetName3, dataSetName4) to override the data set name specified by an output descriptor for RECOVERYDDN for a dynamically allocated output image copy. Use of RECOVERYDSN when the output image copy is not dynamically allocated is an error.
Copy data set ddname construction for JCL-allocated data sets

The copy data set ddnames specified with the REGISTER, OUTCOPYDDN, and RECOVERYDDN options must be coded as follows in the JCL:

- If any of the following conditions exist, the ddnames specified with the REGISTER, OUTCOPYDDN, and RECOVERYDDN options (or the defaults) are coded in the JCL as prefixes to the partition or data set numbers:
  - The object specification is for a specific data set or partition (DSNUM integer specified).
  - The object is partitioned and OUTCOPY BYPART is coded with the OPTIONS command.
  - The object is partitioned and the OUTCOPY installation option value is BYPART.

In the JCL, the ddnames must be in the form namenn or namennnn, where name is the ddname specified with REGISTER, OUTCOPYDDN, or RECOVERYDDN and cannot be more than six characters for objects with less than 100 partitions or five characters for objects with greater than 99 partitions.

For a partitioned table space or index, nnn is the partition number and must be in the range 1 through 999.

**NOTE**

For partitioned spaces with more than 999 partitions, you must use dynamic allocation. You can have 4096 dynamically allocated data sets.

For a nonpartitioned table space or index, nn is the number of a single data set and must be in the range 1 through 32 for table spaces, or 1 through 128 for indexes.

- If the object specification is for all of the data sets of a nonpartitioned object (DSNUM ALL specified) or for all partitions of a partitioned object, and the OUTCOPY option is ASCODED, then the ddnames specified with the REGISTER, OUTCOPYDDN, and RECOVERYDDN options are used “as is” in the JCL. When a request of this type is made, the copy made is for all of the data sets of the object. The ddname cannot be more than eight characters.
Using the specified ddnames in this way ensures that the output copy for each object or partition has a unique ddname. Output copies cannot share data sets, although output copies for multiple object partitions can share the same prefix because the partition number suffix makes each copy DD statement unique.
REBUILD INDEX

This section describes the syntax of the REBUILD INDEX command and its options.

**NOTE**

Considerations:

- When INDEXLOG is set to NO in the installation options or with the OPTIONS command, RECOVER INDEX and REBUILD INDEX are synonyms. Any reference to REBUILD INDEX in this book also applies to RECOVER INDEX when INDEXLOG is set to NO. RECOVER INDEXSPACE, however, is never a synonym for REBUILD INDEX, regardless of the value of INDEXLOG.

- When an index is rebuilt, key values are instantiated in the current version of the table, regardless of the version of the table row from which the key value was extracted. (DB2 Version 8 and later added online schema evolution that provided versioning of tables.)

- When you use REBUILD INDEX with a RECOVER TABLESPACE that includes OBIDXLAT and INDEP OUTSPACE, the OBIDXLAT in the RECOVER TABLESPACE statement must specify the OBIDs for the indexes specified in the REBUILD INDEX statement. Because REBUILD INDEX does not have an OBIDXLAT clause, specifying the OBIDs in the RECOVER TABLESPACE statement is the only way to get the OBIDs translated in the index header page. For more information, see “OBIDXLAT specification” on page 193.

REBUILD INDEX syntax

Figure 26 on page 213 shows the syntax of the REBUILD INDEX command. For all of the conventions used in this diagram, see “Syntax diagrams” on page 24.
Figure 26  REBUILD INDEX syntax diagram

```
REBUILD INDEX
  RECOVER*
  SIMRBLD
  SIMRCVR**
  ALL
  begin: end

OBJECTSET creator.name

INDEP OUTSPACE
  MODEL vcat.BMCDBC.databaseName.indexSpaceName.I0001.znnn
  MODEL userNamedDataSet

TABLESPACE databaseName.
  tableName

INDEP INTABLESPACE
  MODEL vcat.BMCDBC.databaseSame.tableSpaceSame.I0001.znnn
  MODEL userNamedDataSet

PART integer
  begin: end

CLONE
  integer
  page 221

WORKDDN DDName
  DDNamePrefix

NOWORKDDN
  page 222

SORTDEVT deviceType
  page 223

SORTNUM integer
  page 224

NUMREC
  page 224

ANALYZE YES
  page 225
  NO
  ONLY

REDEFINE YES
  page 226
  NO
  NOSCRATCH

REUSE
  page 227

*Is synonymous to REBUILD when INDEXLOG=NO
**Is synonymous to SIMRBLD when INDEXLOG=NO
```
REBUILD INDEX option descriptions

RECOVER PLUS provides the following options for use with the REBUILD INDEX command. The options are described in the order in which they are shown in the REBUILD INDEX syntax diagram.

REBUILD

Use the REBUILD command when you want to rebuild an index from data.

RECOVER

The RECOVER command is synonymous with the REBUILD command when INDEXLOG is set to NO.

NOTE

When you simulate recovery by specifying the SIMULATE YES option, you should use SIMRCVR rather than RECOVER. For more information, see “SIMRCVR” on page 167.

SIMRBLD

SIMRBLD is a synonym for REBUILD. Use the SIMRBLD command to clarify that you are running in simulation mode.

SIMRBLD command generally has the same meaning and the same syntax in simulation mode as a regular REBUILD command has in non-simulation mode.

When you use SIMRBLD, you must also specify OPTIONS SIMULATE YES, which requires a valid Recovery Management for DB2 solution password. For more information, see “SIMULATE YES” on page 134.

SIMRCVR

SIMRCVR is a synonym for SIMRBLD when INDEXLOG is set to NO. More move information, see “SIMRBLD.”
INDEX

The INDEX option specifies the indexes (and the corresponding index spaces) to be rebuilt. You can specify one or more indexes in one command statement.

**NOTE**
RECOVER PLUS does not allow the rebuilding of indexes into DB2 data sets if the corresponding table space recovery is to a non-DB2 (independent) data set or if the corresponding table space recovery is a point-in-time recovery OUTCOPY ONLY.

(indexName,indexName2, . . . . .)

This parameter specifies each of the indexes that you want to rebuild. Each index name is in the form authid.indexName. If you do not provide the authid qualifier, the qualifier defaults to the user ID of the user executing the utility. The list of qualified names must be enclosed in parentheses and separated by commas. You cannot specify any of the indexes on DB2 tables that are used by RECOVER PLUS (such as those on BMCSYNC).

PART integer

This option specifies the partition of the index that you want to rebuild.

**NOTE**
Rebuilding a logical partition is not supported. You must rebuild the entire nonpartitioned index.

PART begin : end

Use this option to recover a range of partitions. For partitioned spaces, begin specifies the number (from 1 through 4095) of the first partition in the range and end specifies the number (from 2 through 4096) of the last partition in the range. The two numbers are separated by a colon (:) with or without spaces. Wrapping partition numbers (for example, DSNUM 4050 : 300) is not supported.

ALL

INDEX(ALL) specifies that all indexes are to be rebuilt for the table space named in the TABLESPACE option.
OBJECTSET creator.name

Use OBJECTSET (REBUILD INDEX OBJECTSET creator.name) to specify that you want to rebuild all of the indexes in a group that was defined using RECOVERY MANAGER.

**NOTE**

SIMRBLD is also valid in place of REBUILD.

The following rules apply to creator.name:

- **creator** specifies the name of the creator of the group and can have up to 128 characters in length.
- **name** species the group name and can have up to 128 characters in length.
- You can delimit both creator and name with single or double quotation marks.
- Both creator and name can contain the special characters $, #, and @ in any position.

RECOVER PLUS uses dynamic grouping to determine the group contents at the time that you run the RECOVER PLUS job containing the OBJECTSET option. Because RECOVER PLUS reads the objects in the group each time the job is executed, objects may be added or removed from the group.

**NOTE**

RECOVER PLUS does not read RECOVERY MANAGER recovery options for the group. If RECOVERY MANAGER recovery options change, you must run RECOVERY MANAGER to pick up the new options and generate the control cards.

Following are restrictions that apply when you use OBJECTSET with REBUILD INDEX:

- AUX is ignored with any specification of OBJECTSET.
- You cannot specify PART with OBJECTSET. RECOVER PLUS uses the group definition for the object.
- You cannot use INDEP OUTSPACE with OBJECTSET.
- You cannot use INDEP INTABLESPACE with OBJECTSET.
- You cannot use TABLESPACE with OBJECTSET.
INDEP OUTSPACE

Use the INDEPENDENT (or INDEP) OUTSPACE option to redirect the output of the rebuilding of an index to data sets other than the data sets that are used by DB2. Whether the indexes being rebuilt are specified by index name or by INDEX(ALL), the output for each index is redirected to a different data set.

You can preallocate the INDEP OUTSPACE data sets using the following guidelines:

- For VCAT-defined spaces, ensure that the volumes or space allocated is large enough to hold the data set.

- For DB2 Version 8 or later, if the source space is defined using a STOGROUP, and if primary and secondary quantities are not defined for the source space, RECOVER PLUS uses a sliding-scale calculation for secondary extents, similar to the method DB2 uses.

For more information, see “Primary space allocation” on page 93 and “Secondary extents” on page 94.

You can allow RECOVER PLUS to dynamically allocate the INDEP OUTSPACE data sets for STOGROUP-defined spaces. If the INDEP OUTSPACE data set does not exist, RECOVER PLUS uses the attributes of the source space to dynamically allocate the target INDEP OUTSPACE data set. If the INDEP OUTSPACE data set exists, RECOVER PLUS uses it as is (no DELETE or DEFINE executed).

RECOVER PLUS includes support for multi-data-set spaces. For example, RECOVER PLUS will dynamically allocate A002, A003, to A00n INDEP OUTSPACE data sets as needed.

To have RECOVER PLUS dynamically allocate the INDEP OUTSPACE data set on a specific volume or volumes, pass a volume list by using the STOGROUP...USEORDER specification in the OPTIONS statement (see page 117).

You can specify the name of each independent data set or you can accept the RECOVER PLUS default.

The default name is identical to the DB2 name of the index space to which the indexes are normally directed, but with BMCDBC substituted for DSNDBC:

$vcat.BMCDBC.databaseName.indexSpaceName.p0001.znnn$

The variables are defined as follows:

- $vcat$ is the ICF catalog name.

- $databaseName.indexSpaceName$ is the name of the database and index space containing the data set being rebuilt.
- $p$ corresponds to the value that is used by DB2 for this node.

- $znnn$ is the data set number.

$z$ represents the letter A, B, C, D, or E. These letters correspond to the first digit—0, 1, 2, 3, or 4—of the partition number. $nnn$ represents the rest of the partition number. $znnn$ represents the partitions numbers as follows:

- A001 through A999 for partitions 1 through 999
- B000 through B999 for partitions 1000 through 1999
- C000 through C999 for partitions 2000 through 2999
- D000 through D999 for partitions 3000 through 3999
- E000 through E096 for partitions 4000 through 4096

When you want to specify a name of your own choosing, use the MODEL option to specify the name of the CLUSTER portion of the VSAM data set that you want to use. The data set name that you specify may include any of the symbolic variables that are listed in the description of the MODEL keyword in the following section.

When you use INDEP OUTSPACE, you must specify it prior to the TABLESPACE specification in the REBUILD INDEX command statement, as shown in Figure 26 on page 213.

If the rebuilding of a partition is performed on a nonpartitioned index, the entire index must be copied to the data set.

If you are rebuilding an index separately from any recovery of its related table space and you specify INDEP INTABLESPACE, you must also specify INDEP OUTSPACE. RECOVER PLUS will not build DB2 index spaces directly from non-DB2 table data sets.

**NOTE**

If you do not specify INDEPENDENT INTABLESPACE when you specify the INDEPENDENT OUTSPACE option with REBUILD INDEX, the table space is stopped and started as READ ONLY (RO).

For information about using redirection with multiple RECOVER PLUS commands, see “Example 9: Recovering to a non-DB2 data set” on page 369. For more information about this feature, see “Testing recovery with simulation” on page 366.

**MODEL**

The MODEL option precedes the non-DB2 data set cluster name to which you want to redirect the output from the rebuilding of the specified index. The data set name that you specify may optionally include any of the symbolic variables in Table 9 on page 180:
TABLESPACE databaseName.tableSpaceName

The TABLESPACE option specifies the table space from which all of the specified indexes are to be rebuilt. If you do not specify the TABLESPACE option, RECOVER PLUS determines the table space for the first valid index specified. All other specified indexes must belong to the same table space. If you specify INDEX(ALL), you must specify TABLESPACE.

- databaseName is the name of the database to which the table space belongs. The name cannot be DSNDB01, DSNDB06, DSNDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE. The default is DSNDB04.

- tableSpaceName is the name of the table space from which the indexes are to be rebuilt. BMCUTIL and BMCSYNC, tables that are used by BMC utilities, cannot be used.

AUX

The AUX option specifies if auxiliary objects will be included with the recovery of the base table spaces. For more information, see “AUX” on page 122.

INDEP INTABLESPACE

Use the INDEPENDENT (or INDEP) INTABLESPACE option to specify that the indexes to be rebuilt are to be extracted from a data set other than the table space data set that is used by DB2.

If the independent data set is one of your own choosing that you previously created, you must specify the name of that data set with the MODEL option. If you do not specify a name, RECOVER PLUS assumes the RECOVER PLUS data set name model to construct the data set name:

\[(vcat.BMCDBC.databaseName.tableSpaceName.p0001.znnn)\]

The variables are defined as follows:

- vcat is the ICF catalog name.

- databaseName.tableSpaceName is the name of the database and space containing the data set being rebuilt.

- p corresponds to the value that is used by DB2 for this node.

- znnn is the data set number.
z represents the letter A, B, C, D, or E. These letters correspond to the first digit—0, 1, 2, 3, or 4—of the partition number. nnn represents the rest of the partition number. znnn represents the partitions numbers as follows:

- A001 through A999 for partitions 1 through 999
- B000 through B999 for partitions 1000 through 1999
- C000 through C999 for partitions 2000 through 2999
- D000 through D999 for partitions 3000 through 3999
- E000 through E096 for partitions 4000 through 4096

If you specify a RECOVER TABLESPACE command (for the table space associated with this index) by using INDEP OUTSPACE in the same SYSIN stream as this REBUILD INDEX command and you also specify INDEP INTABLESPACE, any data set name used must be the same as the name used with INDEP OUTSPACE in the RECOVER TABLESPACE command. If you do not include a data set name, RECOVER PLUS assumes the data set that is named in INDEP OUTSPACE. If you also do not specify INDEP INTABLESPACE, RECOVER PLUS still uses the independent table space data set that is named in the RECOVER TABLESPACE command to extract the keys.

When you use INDEP INTABLESPACE, you must specify it immediately after the TABLESPACE specification, as shown in Figure 26 on page 213.

For information about using redirection with multiple RECOVER PLUS commands, see “Example 9: Recovering to a non-DB2 data set” on page 369.

**MODEL**

Use the MODEL option following INDEP INTABLESPACE to specify an existing non-DB2 data set from which the indexes are to be rebuilt. This data set was previously used to receive the output from the recovery of the associated table space.

**PART**

The PART option specifies the partitions of an index on a partitioned table space to rebuild. You can specify the index partition individually in the main body of the REBUILD INDEX command or in a list by using the INDEX(indexName1, indexName2...) option, but you cannot specify the partition in both places. If you specify the partition by using the REBUILD INDEX command statement, you can place the partition specification before or after the TABLESPACE option.

**PART ALL**

PART ALL is the default and tells RECOVER PLUS to rebuild all partitions of the index space.
PART integer

PART integer tells RECOVER PLUS to rebuild the index space for the specified partition. If you specify a nonpartitioned index, the index entries are replaced for the named partition.

integer must be in the range 1 through 4096.

PART begin : end

Use this option to recover a range of partitions for the index space. For partitioned index spaces, begin specifies the number (from 1 through 4095) of the first partition in the range and end specifies the number (from 2 through 4096) of the last partition in the range. The two numbers are separated by a colon (:) with or without spaces. Wrapping partition numbers (for example, DSNUM 4050 : 300) is not supported.

CLONE

The CLONE option indicates that RECOVER PLUS is to rebuild clone index data. When you work with clone data, the following limitations apply:

- You cannot refer to the same object with the CLONE option and without the CLONE option in the same recovery step; you cannot process the base and its clone in the same command.

- Related objects (table space and all indexes for its table, all related LOB objects) should use the same CLONE specification in the same recovery step.

WORKDDN DDName

The WORKDDN DDName option specifies the work data set ddname. The default is SYSUT1.

When you use multiple REBUILD INDEX command statements, the following restrictions apply to the use of WORKDDN:

- If you use multiple command statements for a nonpartitioned table space, the DDName that you specify for WORKDDN must be the same in all of the statements.

- If you use multiple command statements for a partitioned table space and request the rebuilding of a nonpartitioned index, the DDName that you specify for WORKDDN must be the same in all of the statements. Otherwise, RECOVER PLUS cannot extract keys for all indexes from each partition in one pass of the table space.
If you specify only the rebuilding of a partitioned index for a partitioned table space with multiple REBUILD INDEX command statements, the value specified for WORKDDN has no restrictions.

WORKDDN can not be specified for a partitioned index if you use RECOVER UNLOADKEYS for the space.

**NOTE**

RECOVER PLUS ignores WORKDDN if the value of the MAXKSORT option is greater than 1. The value of MAXKSORT is set in the installation options or on the OPTIONS command. For more information about MAXKSORT, see page 136 and page 544.

For more information, see “Restrictions for key work data sets, SKEYDDN, SORTNUM, and SORTDEVT” on page 264.

**NOWORKDDN**

The NOWORKDDN option causes the extracted keys to be sent directly to the sort routine without first writing these keys to a work data set. Using this option causes the index to rebuild faster. However, restartability is more limited because, without the work data set, restarting at the beginning of the BUILD phase is not possible. The restart will have to redo the UNLOAD phase.

The keys are sent directly to the sort if neither WORKDDN nor NOWORKDDN is specified, and no SYSUT1 DD statement exists in the JCL.

If you use RECOVER UNLOADKEYS for the space, NOWORKDDN is the default for a partitioned index.

**SORTDEVT and SORTNUM sort work data set options**

To sort index keys, RECOVER PLUS invokes BMCSORT, which allocates the required temporary sort work data sets. You can take one of the following courses of action:

- Specify SORTNUM and SORTDEVT to direct the allocation.

- Specify neither SORTNUM nor SORTDEVT, and let BMCSORT allocate the work data sets according to sort rules.

- Specify neither SORTNUM nor SORTDEVT, and specify SxxxWKnn DD statements in the JCL.

- Specify only SORTDEVT, and let BMCSORT determine the number of data sets.
If you use SxxxWKnn DD statements in the JCL, any SORTDEVT and SORTNUM specifications that are present are ignored. If you do not use SxxxWKnn DD statements and do not specify SORTDEVT or SORTNUM, the sort routine uses the installation options for SORTDEVT and SORTNUM.

When you use multiple REBUILD INDEX command statements, the following restrictions apply to the use of SORTNUM and SORTDEVT:

- If you use multiple command statements for a nonpartitioned table space, the values that you use for SORTNUM and SORTDEVT must be the same in all of the statements because RECOVER PLUS unloads the keys concurrently for all of the indexes.

- If you use multiple command statements for a partitioned table space and you specify a nonpartitioned index recovery, the values that you use for SORTNUM and SORTDEVT must be the same in all of the statements so RECOVER PLUS can extract keys for all indexes from each partition in one pass of the table space.

If you specify the rebuilding of a partitioned index for a partitioned table space with multiple REBUILD INDEX command statements, the values specified for SORTNUM and SORTDEVT have no restrictions.

**SORTDEVT deviceType**

SORTDEVT deviceType specifies the device type for the temporary sort work data sets that BMCSORT uses for sorting the index.

If you specify SORTDYN NO (page 119), BMCSORT defaults are used even if you specify a value for SORTDEVT. Otherwise, RECOVER PLUS uses its internal default value, SYSDA, if you do not specify SORTDEVT.

**SORTNUM integer**

The SORTNUM option affects the allocation of sort work files when BMCSORT is allocating your sort work files dynamically. You can specify an integer value of 1 through 255.

When you specify this option, BMCSORT dynamically allocates the number of sort work files that it needs for each sort task up to the maximum that is illustrated in the following formula:

\[
\text{maximum dynamically allocated sort work files} = n - \text{preallocated sort work files}
\]

If you specify integer from 1 through 32, \(n\) equals 32.

If you specify integer greater than 32, \(n\) equals integer.
NOTE
Preallocated sort work files include sort work files that are allocated in your JCL.

If you do not specify a value for SORTNUM, RECOVER PLUS uses the installation option default.

NUMREC

NUMREC specifies the size of the sort file for the index sort routine. This option is appropriate only when NOWORKDDN is specified because the utility counts records when a work data set is used to collect the keys before the sort begins. The size is defined as an estimate or the exact number of key records to process. For information about performance implications, see Chapter 6, “RECOVER PLUS concepts.”

TIP
RECOVER PLUS usually determines a good estimate for the number of key records automatically. BMC recommends that in most cases you do not specify NUMREC or that you specify NUMREC CALC (the default).

NUMREC CALC

NUMREC CALC is the default value.

Specifying this option causes RECOVER PLUS to estimate the number of key records to be sorted. RECOVER PLUS uses RUNSTATS statistics, if available, to make these estimates. If these statistics are not available, RECOVER PLUS tries to estimate the number of key records based on the number of pages in the associated table space.

RECOVER PLUS uses a minimum estimate of 400,000 for each index key sort (or 200,000 if the key length is greater than 1000 characters).

NUMREC NOEST

When you specify NUMREC NOEST, RECOVER PLUS passes an estimate of 400,000 to each index key sort (or 200,000 if the key length is greater than 1000 characters).

NUMREC EST integer

Use NUMREC EST integer to specify an estimated number of key records for the index sort routine. integer must be a positive integer.
If you have specified multiple indexes on the REBUILD or RECOVER UNLOADKEYS statement, RECOVER PLUS divides the NUMREC value equally among the indexes. In general, NUMREC EST is unnecessary and might degrade performance unless the specified value is accurate.

If the specified estimate is less than 400,000 (200,000 for keys with a length greater than 1000 characters), RECOVER PLUS uses an estimate of 400,000 (200,000 if the key length is greater than 1000 characters).

**NUMREC ABS integer**

Use NUMREC ABS integer to specify the exact number of key records for the index sort routine. integer must be a positive integer.

---

**WARNING**

If integer is a value that does not represent the true number of key records passed to the index sort routine, the index sort routine abends. Use extreme caution when you specify NUMREC ABS.

**ANALYZE**

The ANALYZE option prints the index rebuilding plan before executing that plan. Information about the following items used for rebuilding the index is included in the plan:

- the record sizes for index sort work data sets
- the phases to occur during execution
- the steps to occur within each phase

Use this information to allocate space more accurately for index sort work data sets and index key work data sets, thereby limiting abends that are caused by inadequate data set allocations. You can use these statistics with historical information about the times required to perform the various operations to estimate the time required to rebuild the indexes.

---

**NOTE**

If you coded ANALYZE with the OPTIONS command, you must code the same value here. If you use multiple REBUILD INDEX command statements you must use the same value for the ANALYZE option in all of the statements. If you specify a value for ANALYZE on one command statement and take the default on the others, the value that you specify on the single command statement becomes the default for the others.
ANALYZE YES

ANALYZE YES is the default and provides all of the information described in the preceding section.

ANALYZE NO

If you specify ANALYZE NO, RECOVER PLUS provides no information.

ANALYZE ONLY

If you specify ANALYZE ONLY, RECOVER PLUS provides the same information as it provides with ANALYZE YES and then stops the job when the ANALYZE phase is complete. You can use the data from the ANALYZE phase to determine the resources required and what will happen during the recovery job. You cannot restart a RECOVER PLUS job that specifies ANALYZE ONLY, but you can start a new recovery job.

Use the ANALYZE ONLY option to determine which phases will occur with a specific request, or which copy data sets, log data sets, and change accumulation data sets are used in the rebuilding of the indexes.

REDEFINE

Use the REDEFINE option to avoid the normal deletion and reallocation of the index space data sets for STOGROUP-defined index spaces. This option is ignored for VCAT-defined index space data sets.

The value that you specify for the REDEFINE option applies to all STOGROUP-defined index spaces included in a single REBUILD INDEX command statement. To rebuild multiple STOGROUP-defined index spaces with some reallocated and others not reallocated, you must use one REBUILD INDEX command statement for the index spaces to be reallocated and a second REBUILD INDEX command statement for those index spaces not to be reallocated.

Regardless of the value of REDEFINE, if the data sets for a STOGROUP-defined index space do not exist, RECOVER PLUS creates the data sets.

NOTE

When you are rebuilding a multi-data-set, nonpartitioned index space, more or fewer data sets may result than were previously used.
REDEFINE YES

REDEFINE YES is the default. If you specify REDEFINE YES, RECOVER PLUS deletes and reallocates the VSAM data sets for STOGROUP-defined index spaces prior to the start of the BUILD phase. For more information, see “Storage group-defined data sets” on page 92.

NOSCRATCH

If you specify NOSCRATCH with REDEFINE YES, RECOVER PLUS issues IDCAMS to uncatalog without scratching. You can use NOSCRATCH to support disaster recovery scenarios where the following items are true:

- DASD volumes at the recovery site differ from those at the primary site.
- The operating system catalog and the DB2 catalog and directory are restored before running RECOVER PLUS.
- DB2 STOGROUPS are changed to point to the recovery site packs.

Using this option eliminates mount messages for the primary site packs that do not exist at the recovery site.

NOTE

REDEFINE YES has no effect if you specify INDEP OUTSPACE. For the instructions for creating data sets for INDEP OUTSPACE, see “INDEP OUTSPACE” on page 217.

REDEFINE NO

REDEFINE NO tells RECOVER PLUS not to delete and reallocate the VSAM data sets for the index space. Use this option to treat STOGROUP-defined index spaces as VCAT-defined index spaces and to reallocate the VSAM data sets manually. New volumes are used as needed from the STOGROUP, even with REDEFINE NO specified.

REUSE

The REUSE option specifies that the space will not be deleted or redefined. This option cannot be used with the REDEFINE option.

NOCOPYPEND

If you specify the NOCOPYPEND option, RECOVER PLUS resets ICOPY status on COPY YES indexes and issues message BMC96232 to inform you that ICOPY status has been reset even though the space is not recoverable.
RECOVER UNLOADKEYS

This section describes the syntax of the RECOVER UNLOADKEYS command and its options.

**NOTE**

RECOVER UNLOADKEYS ignores the MAXKSORT option and processes as if you specified MAXKSORT=1. If you also specify a REBUILD INDEX for the partitioning index and specify a MAXKSORT value of at least 3, RECOVER PLUS uses a separate sort for the partitioning index and does the REBUILD INDEX for the partitioning index in parallel with RECOVER UNLOADKEYS.

RECOVER UNLOADKEYS syntax

Figure 27 shows the syntax of the RECOVER UNLOADKEYS command. For the conventions used in this diagram, see “Syntax diagrams” on page 24.

Figure 27  RECOVER UNLOADKEYS syntax diagram
RECOVER PLUS provides the following options for use with the RECOVER UNLOADKEYS command. The options are described in the order in which they are shown in the RECOVER UNLOADKEYS syntax diagram.

**RECOVER**

Use the RECOVER command when you want to perform an actual recovery.

**SIMRCVR**

SIMRCVR is a synonym for the RECOVER command. Use the SIMRCVR command to clarify that you are running in simulation mode. When you use SIMRCVR, you must also specify OPTIONS SIMULATE YES, which requires a valid Recovery Management for DB2 solution password. For more information, see “SIMULATE YES” on page 134.

**UNLOADKEYS**

UNLOADKEYS specifies one or more indexes for which keys and row IDs are to be extracted. All specified indexes must be nonpartitioned indexes on the table of a partitioned table space.

(indexName1,indexName2,...)

This specifies each of the indexes for which keys and row IDs are to be extracted. Each index name is in the form authid.indexName. If you do not provide the authid qualifier, the qualifier is defaulted to the user ID of the user executing the utility. You must enclose the list of qualified names in parentheses.

---

**NOTE**

RECOVER UNLOADKEYS does not support indexes defined using an expression.

**PART integer**

This parameter specifies the partition of the index for which keys and row IDs are to be extracted.
**PART begin : end**

Use this option to recover a range of partitions for the index space. For partitioned index spaces, `begin` specifies the number (from 1 through 4095) of the first partition in the range and `end` specifies the number (from 2 through 4096) of the last partition in the range. The two numbers are separated by a colon (:) with or without spaces. Wrapping partition numbers (for example, DSNUM 4050 : 300) is not supported.

**ALL**

UNLOADKEYS(ALL) specifies that keys and row IDs are to be extracted for all nonpartitioned indexes on the table space named in the TABLESPACE option.

**TABLESPACE databaseName.tableSpaceName**

The TABLESPACE option specifies the partitioned table space from which keys and row IDs are to be extracted for the nonpartitioned indexes specified by the UNLOADKEYS option. If you specify UNLOADKEYS(ALL), you must also specify TABLESPACE. If you specify UNLOADKEYS(indexName1, . . . . .) and do not specify TABLESPACE, RECOVER PLUS determines the table space for the first index specified. All other specified indexes must belong to the same table space.

- `databaseName` is the name of the database to which the table space belongs. The default is DSNDB04. The name cannot be DSNDB01, DSNDB06, DSNDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE.

- `tableSpaceName` is the name of the table space from which nonpartitioned index keys are to be extracted. BMCUTIL and BMCSYNC, tables that are used by BMC utilities, cannot be used.

**PART**

The PART option specifies the partitions of an index on a partitioned table space to unload. You can specify the index partition individually in the main body of the RECOVER UNLOADKEYS command or in a list by using the INDEX(indexName1, indexName2...) option, but you cannot specify the partition in both places. If you specify the partition by using the RECOVER UNLOADKEYS command statement, you can place the partition specification before or after the TABLESPACE option.

**PART integer**

PART integer specifies the partition of the table space from which the keys and row IDs are extracted. You must always specify this option with RECOVER UNLOADKEYS.

`integer` must be in the range 1 through 4096.
You can place the partition specification before or after the TABLESPACE option. You can specify a partition individually in the main body of the RECOVER UNLOADKEYS command statement or in a list by using the UNLOADKEYS(indexName1, indexName2...) option, but you cannot specify the partition in both places.

**PART begin : end**

Use this option to recover a range of partitions for the index space. For partitioned index spaces, begin specifies the number (from 1 through 4095) of the first partition in the range and end specifies the number (from 2 through 4096) of the last partition in the range. The two numbers are separated by a colon (:) with or without spaces. Wrapping partition numbers (for example, DSNUM 4050 : 300) is not supported.

**INDEPENDENT INTABLESPACE**

Use the INDEPENDENT (or INDEP) INTABLESPACE option to specify that the nonpartitioned index keys and row IDs are to be extracted from a table space data set other than the one that is used by DB2.

If the independent data set is one of your own choosing that you previously created, you must specify the name of that data set with the MODEL option. If you do not specify a name, RECOVER PLUS assumes the RECOVER PLUS data set name model to construct the data set name:

\[(\text{vcat}.\text{BMCDBC.}\text{databaseName.indexSpaceName.p0001.nnn})\]

The variables are defined as follows:

- **vcat** is the ICF catalog name.
- **databaseName.indexSpaceName** is the name of the database and index space containing the data set being rebuilt.
- **p** corresponds to the value that is used by DB2 for this node.
- **znnn** is the data set number.

\(z\) represents the letter A, B, C, D, or E. These letters correspond to the first digit—0, 1, 2, 3, or 4—of the partition number. \(nnn\) represents the rest of the partition number. \(znnn\) represents the partitions numbers as follows:

- A001 through A999 for partitions 1 through 999
- B000 through B999 for partitions 1000 through 1999
- C000 through C999 for partitions 2000 through 2999
- D000 through D999 for partitions 3000 through 3999
- E000 through E096 for partitions 4000 through 4096
If you specify a RECOVER TABLESPACE command (for the table space associated with this index) by using INDEP OUTSPACE in the same SYSIN stream as this RECOVER UNLOADKEYS command and you specify INDEP INTABLESPACE, any data set name specified must be the same as the name used with INDEP OUTSPACE in the RECOVER TABLESPACE command. If you do not include a data set name, RECOVER PLUS assumes that data set that is named in INDEP OUTSPACE. If you also do not specify INDEP INTABLESPACE, RECOVER PLUS still uses the independent table space data set that is named in the RECOVER TABLESPACE command to extract the keys.

When you use INDEP INTABLESPACE, you must specify it immediately after the TABLESPACE specification, as shown in Figure 27 on page 228.

For information about using redirection with multiple RECOVER PLUS commands, see “Example 9: Recovering to a non-DB2 data set” on page 369. For more information about some applications of this feature, see “Testing recovery with simulation” on page 366.

MODEL

Use the MODEL option following INDEP INTABLESPACE to specify an existing non-DB2 data set from which the nonpartitioned index keys and row IDs are to be extracted. This non-DB2 data set was previously used to receive the output from the recovery of the associated table space.

CLONE

The CLONE option indicates that RECOVER PLUS is to recover clone index data. When you work with clone data, the following limitations apply:

- You cannot refer to the same object with the CLONE option and without the CLONE option in the same recovery step; you cannot process the base and its clone in the same command.

- Related objects (table space and all indexes for its table, all related LOB objects) should use the same CLONE specification in the same recovery step.

SKEYDDN

Use the SKEYDDN option to specify the ddname for the unloaded keys work data set that is to be used to hold the sorted keys and row IDs. The default ddname is SKEY. The maximum length of the ddname specified by SKEYDDN is eight characters. Certain restrictions apply to this option, as follows:

- If you use multiple RECOVER UNLOADKEYS command statements for a table space, the ddname that you specify for SKEYDDN must be the same in all of the statements.
If you use other RECOVER UNLOADKEYS command statements for other table spaces or other REBUILD INDEX command statements for other table spaces, any value you specify for SKEYDDN with those statements must be different.

If you use RECOVER UNLOADKEYS and REBUILD INDEX command statements for a table space, REBUILD INDEX defaults to NOWORKDDN if you specify SKEYDDN.

Do not use data sets created with SIMRCVR UNLOADKEYS except with SIMRCVR BUILDINDEX.

**SORTDEV and SORTNUM sort work data set options**

To sort index keys, RECOVER PLUS invokes BMCSORT, which allocates the required temporary sort work data sets. You can take one of the following courses of action:

- Specify SORTNUM and SORTDEV to direct the allocation.
- Specify neither SORTNUM nor SORTDEV, and let BMCSORT allocate the work data sets according to sort rules.
- Specify neither SORTNUM nor SORTDEV, and specify SxxxWKnn DD statements in the JCL.
- Specify only SORTDEV, and let BMCSORT determine the number of data sets.

If you use SxxxWKnn (sort work) DD statements in the JCL, any SORTDEV and SORTNUM specifications that are present are ignored. If you do not use SxxxWKnn DD statements and do not specify SORTDEV or SORTNUM, the sort routine uses the installation options for SORTDEV or SORTNUM.

**SORTDEV**

devicetype

SORTDEV devicetype specifies the device type for the temporary sort work data sets that BMCSORT uses when sorting the index keys.

If you specify SORDYNO (page 119), BMCSORT defaults are used even if you specify a value for SORTDEV. Otherwise, RECOVER PLUS uses its internal default value, SYSDA, if you do not specify SORTDEV.

**SORTNUM**

integer

The SORTNUM option affects the allocation of sort work files when BMCSORT is allocating your sort work files dynamically. You can specify an integer value of 1 through 255.
When you specify this option, BMCSORT dynamically allocates the number of sort work files that it needs for each sort task up to the maximum that is illustrated in the following formula:

\[
\text{maximum dynamically allocated sort work files} = n - \text{preallocated sort work files}
\]

If you specify \textit{integer} from 1 through 32, \( n \) equals 32. If you specify \textit{integer} greater than 32, \( n \) equals integer.

\textbf{NOTE}

Preallocated sort work files include sort work files that are allocated in your JCL.

If you do not specify a value for \textit{SORTNUM}, RECOVER PLUS uses the installation option default.

\textbf{NUMREC}

You must specify the same \textit{NUMREC} values for all \textit{RECOVER UNLOADKEYS} command statements for the same table space.

\textit{NUMREC} specifies the size of the sort file for the index sort routine. The size is defined as an estimate or the exact number of key records to process. For information about performance implications, see Chapter 6, “RECOVER PLUS concepts.”

\textbf{TIP}

RECOVER PLUS usually determines a good estimate for the number of key records automatically. BMC recommends that in most cases you do not specify \textit{NUMREC} or that you specify \textit{NUMREC CALC} (the default).

\textbf{NUMREC CALC}

\textit{NUMREC CALC} is the default value.

Specifying this option allows RECOVER PLUS to estimate the number of key records to be sorted. RECOVER PLUS uses RUNSTATS statistics, if available, to make these estimates. If these statistics are not available, RECOVER PLUS tries to estimate the number of key records based on the number of pages in the associated table space.

RECOVER PLUS uses a minimum estimate of 400,000 for each index key sort (or 200,000 if the key length is greater than 1000 characters).
NUMREC NOEST

When you specify NUMREC NOEST, RECOVER PLUS passes an estimate of 400,000 to each index key sort (or 200,000 if the key length is greater than 1000 characters).

NUMREC EST integer

Use NUMREC EST integer to specify an estimated number of key records for the index sort routine. integer must be a positive integer.

If you have specified multiple indexes on the REBUILD or RECOVER UNLOADKEYS statement, RECOVER PLUS divides the NUMREC value equally among the indexes. In general, NUMREC EST is unnecessary and might degrade performance unless the specified value is accurate.

If the specified estimate is less than 400,000 (200,000 for keys with a length greater than 1000 characters), RECOVER PLUS uses an estimate of 400,000 (200,000 if the key length is greater than 1000 characters).

NUMREC ABS integer

Use NUMREC ABS integer to specify the exact number of key records for the index sort routine. integer must be a positive integer.

**WARNING**

If integer is a value that does not represent the true number of key records passed to the index sort routine, the index sort routine will abend. Use extreme caution when you specify NUMREC ABS.

ANALYZE

The ANALYZE option prints a recovery plan before executing that plan. Information about the following items used for recovery is included in the plan:

- the phases to occur during execution
- the steps to occur within each phase

Use this information to allocate space more accurately for the unload work data sets so limiting abends that are caused by inadequate data set allocations. You can use these statistics with historical information about the times required to perform the various operations to estimate recovery time.
**NOTE**

If you coded ANALYZE with the OPTIONS command, you must code the same value here. If you use multiple RECOVER PLUS command statements, you must use the same value for the ANALYZE option in all of the statements. If you specify a value for ANALYZE on one command statement and take the default on the others, the value you specify on the single command statement becomes the default for the others.

**ANALYZE YES**

ANALYZE YES is the default and provides information about the items used for recovery that are in the recovery plan.

**ANALYZE NO**

If you specify ANALYZE NO, RECOVER PLUS provides no information.

**ANALYZE ONLY**

If you specify ANALYZE ONLY, RECOVER PLUS provides the same information as it provides with ANALYZE YES and then stops the job when the ANALYZE phase is complete. You can use the data from the ANALYZE phase to determine the resources required and what will happen during the recovery job. You cannot restart a RECOVER PLUS job that specifies ANALYZE ONLY, but you can start a new recovery job.

Use the ANALYZE ONLY option to determine which phases will occur with a specific recovery request.
RECOVER BUILDINDEX

This sections describes the syntax of the RECOVER BUILDINDEX command and its options.

**NOTE**

RECOVER BUILDINDEX ignores the MAXKSORT option and processes as if you specified MAXKSORT=1.

RECOVER BUILDINDEX syntax

Figure 28 shows the syntax of the RECOVER BUILDINDEX command. For the conventions used in this diagram, see “Syntax diagrams” on page 24.

Figure 28  RECOVER BUILDINDEX syntax diagram
RECOVER BUILDINDEX option descriptions

RECOVER PLUS provides the following options for use with the RECOVER BUILDINDEX command. The options are described in the order in which they are shown in the RECOVER BUILDINDEX syntax diagram.

RECOVER

Use the RECOVER command when you want to perform an actual recovery.

SIMRCVR

SIMRCVR is a synonym for the RECOVER command. Use the SIMRCVR keyword to clarify that you are running in simulation mode. When you use SIMRCVR, you must also specify OPTIONS SIMULATE YES, which requires a valid Recovery Management for DB2 solution password. For more information, see “SIMULATE YES” on page 134.

BUILDINDEX

The BUILDINDEX option specifies one or more nonpartitioned indexes (and so identifies the corresponding index spaces) to be rebuilt. The keys for the indexes must have already been extracted in RECOVER UNLOADKEYS jobs and be available in the work data sets (the unload data sets) specified in that job.

(indexName1,indexName2, ... )

This parameter specifies each of the nonpartitioned indexes that you want to rebuild. Each index name is in the form authid.indexName. If you do not provide the authid qualifier, the qualifier is defaulted to the user ID of the user executing the utility. The keys and row IDs for all of the indexes must be already contained in the unload data sets specified in the job.

You must enclose the list of qualified names in parentheses.

ALL

BUILDINDEX(ALL) specifies that all nonpartitioned indexes on the table space are to be rebuilt and expects that the keys and row IDs are contained in the unload data sets specified in the job.
INDEPENDENT OUTSPACE

Use the INDEPENDENT (or INDEP) OUTSPACE option to redirect the output of a RECOVER BUILDINDEX run to data sets other than those that are used by DB2.

You can preallocate the INDEP OUTSPACE data sets using the following guidelines:

- For VCAT-defined spaces, ensure that the volumes or space allocated is large enough to hold the data set.

- For DB2 Version 8 or later, if the source space is defined using a STOGROUP, and if primary and secondary quantities are not defined for the source space, RECOVER PLUS uses a sliding-scale calculation for secondary extents, similar to the method DB2 uses.

For more information, see “Primary space allocation” on page 93 and “Secondary extents” on page 94.

You can allow RECOVER PLUS to dynamically allocate the INDEP OUTSPACE data sets for STOGROUP-defined spaces. If the INDEP OUTSPACE data set does not exist, RECOVER PLUS uses the attributes of the source space to dynamically allocate the target INDEP OUTSPACE data set. If the INDEP OUTSPACE data set exists, RECOVER PLUS uses it as is (no DELETE or DEFINE executed).

RECOVER PLUS includes support for multi-data-set spaces. For example, RECOVER PLUS will dynamically allocate A002, A003, to A00n INDEP OUTSPACE data sets as needed.

To have RECOVER PLUS dynamically allocate the INDEP OUTSPACE data set on a specific volume or volumes, pass a volume list by using the STOGROUP...USEORDER specification in the OPTIONS statement (see page 117).

You can specify the name of each independent data set or you can accept the RECOVER PLUS default.

The default name is identical to the DB2 name of the index space to which the indexes are normally directed but with BMCDBC substituted for DSNDBC:

\[ \text{vcat.BMCDBC.databaseName.indexSpaceName.p0001.znnn} \]

The variables are defined as follows:

- \( \text{vcat} \) is the ICF catalog name.

- \( \text{databaseName.indexSpaceName} \) is the name of the database and index space containing the data set being rebuilt.

- \( p \) corresponds to the value that is used by DB2 for this node.
■ znnn is the data set number.

z represents the letter A, B, C, D, or E. These letters correspond to the first digit—0, 1, 2, 3, or 4—of the partition number. nnn represents the rest of the partition number. znnn represents the partitions numbers as follows:

■ A001 through A999 for partitions 1 through 999
■ B000 through B999 for partitions 1000 through 1999
■ C000 through C999 for partitions 2000 through 2999
■ D000 through D999 for partitions 3000 through 3999
■ E000 through E096 for partitions 4000 through 4096

Any data set name that you specify must be preceded by the MODEL keyword.

When you want to specify a name of your own choosing, use the MODEL option to specify the name of the CLUSTER portion of the VSAM data set that you want to use. The data set name that you specify may include any of the symbolic variables that are listed in the description of the MODEL keyword in the following section.

When you use INDEP OUTSPACE, you must specify it prior to the TABLESPACE specification in the RECOVER BUILDINDEX command statement, as shown in Figure 28 on page 237.

For information about using redirection with multiple RECOVER PLUS commands, see “Example 9: Recovering to a non-DB2 data set” on page 369. For information about some applications of this feature, see “Testing recovery with simulation” on page 366.

MODEL

The MODEL option precedes the non-DB2 data set cluster name to which you want to redirect the output from the rebuilding of the specified index. The data set name that you specify may optionally include any of the symbolic variables in Table 9 on page 180:

TABLESPACE databaseName.tableSpaceName

The TABLESPACE option specifies the partitioned table space from which the index keys and row IDs contained in the unload data sets were extracted. If you specify BUILDINDEX(ALL), you must also specify TABLESPACE. If you specify BUILDINDEX(indexName1,. . . . . ) and do not specify TABLESPACE, RECOVER PLUS determines the table space for the first index specified. All other specified indexes must belong to that table space.

■ databaseName is the name of the database to which the table space belongs. The default is DSNDB04. The name cannot be DSNDB01, DSNDB06, DSNDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE.
- `tableSpaceName` is the name of the table space for which all nonpartitioned indexes are to be rebuilt. BMCUTIL and BMCSYNC, tables that are used by BMC utilities, cannot be used.

**CLONE**

The CLONE option indicates that RECOVER PLUS is to recover clone index data. When you work with clone data, the following limitations apply:

- You cannot refer to the same object with the CLONE option and without the CLONE option in the same recovery step; you cannot process the base and its clone in the same command.

- Related objects (table space and all indexes for its table, all related LOB objects) should use the same CLONE specification in the same recovery step.

**SKEYDDN**

The SKEYDDN option specifies a prefix for the ddnames of the unloaded keys work data sets created by prior RECOVER UNLOADKEYS job steps. The unload work data sets must contain all of the keys needed to rebuild all of the indexes requested. Any ddname within the JCL beginning with the specified prefix is recognized as an unload work data set. The remainder of the ddname is not restricted to numeric values. However, the maximum length of the prefix specified by SKEYDDN is eight characters. The default value is SKEY.

Do not use data sets created with SIMRCVR UNLOADKEYS except with SIMRCVR BUILDINDEX.

**ANALYZE**

The ANALYZE option prints a recovery plan before executing that plan. Information about the following items used for recovery is included in the plan:

- the record sizes for index sort work data sets
- the phases to occur during execution
- the steps to occur within each phase

Use this information to allocate space more accurately for index sort work data sets and index key work data sets, thereby limiting abends that are caused by inadequate data set allocations. You can use these statistics with historical information about the times required to perform the various operations to estimate recovery time.
ANALYZE YES

ANALYZE YES is the default and provides all of the information in the recovery plan.

ANALYZE NO

If you specify ANALYZE NO, RECOVER PLUS provides no information.

ANALYZE ONLY

If you specify ANALYZE ONLY, RECOVER PLUS provides the same information as it provides with ANALYZE YES and then stops the job when the ANALYZE phase is complete. You can use the data from the ANALYZE phase to determine the resources required and what will happen during the recovery job. You cannot restart a RECOVER PLUS job that specifies ANALYZE ONLY, but you can start a new recovery job.

Use the ANALYZE ONLY option to determine which phases will occur with a specific recovery request.

REDEFINE

Use the REDEFINE option to avoid the normal deletion and reallocation of the index space data sets for STOGROUP-defined index spaces. This option is ignored for VCAT-defined index space data sets.

The value that you specify for REDEFINE applies to all STOGROUP-defined index spaces included in a single RECOVER BUILDINDEX command statement. To rebuild multiple STOGROUP-defined index spaces with some reallocated and others not reallocated, you must use one RECOVER BUILDINDEX command statement for the index spaces to be reallocated and a second RECOVER BUILDINDEX command statement (in a separate execution of RECOVER PLUS) for those index spaces not to be reallocated.

Regardless of the value of REDEFINE, if the data sets for a STOGROUP-defined index space do not exist, RECOVER PLUS allocates the VSAM data sets for the index space.

NOTE

When a RECOVER BUILDINDEX of a multi-data-set index space is performed, more or fewer data sets may result than were previously used.
REDEFINE YES

REDEFINE YES is the default. If you specify REDEFINE YES, RECOVER PLUS deletes and reallocates the VSAM data sets for STOGROUP-defined index spaces prior to the start of the BUILD phase. For more information, see “Storage group-defined data sets” on page 92.

NOSCRATCH

If you specify NOSCRATCH with REDEFINE YES, RECOVER PLUS issues IDCAMS to uncatalog without scratching. You can use NOSCRATCH to support disaster recovery scenarios where the following items are true:

- The DASD volumes at the recovery site differ from those at the primary site.
- The operating system catalog and the DB2 catalog and directory are restored prior to running RECOVER PLUS.
- DB2 STOGROUPS are changed to point to the recovery site packs.

Using this option eliminates mount messages for the primary site packs that do not exist at the recovery site.

NOTE

REDEFINE YES has no effect if INDEP OUTSPACE is coded. For instructions for creating data sets for INDEP OUTSPACE, see “INDEPENDENT OUTSPACE” on page 239.

REDEFINE NO

REDEFINE NO tells RECOVER PLUS not to delete and reallocate the VSAM data sets for the index space. Use this option to treat STOGROUP-defined index spaces as VCAT-defined index spaces and to reallocate the VSAM data sets manually. New volumes are used as needed from the STOGROUP, even with REDEFINE NO specified.

REUSE

The REUSE option specifies that the space will not be deleted or redefined. This option cannot be used with the REDEFINE option.
This section describes the syntax of the LOGSCAN command and its options.

**LOGSCAN syntax**

Figure 29 through Figure 33 on page 246 show the syntax of the LOGSCAN command. For the conventions used in this diagram, see “Syntax diagrams” on page 24.

*Not valid with BACKOUT, which is part of the Point-in-Time Recovery Specification*
**Figure 30** LOGSCAN syntax—Table space specification

### Table space specification

- `TABLESPACE` page 248
- `tableSpaceName`
- `databaseName`
- `OBJECTSET` page 255
- `creator.name`
- `INDEXES`
- `INDEX` page 249
  - `YES` / `NO`
- `INDEX`
- `FROMRBA`
- `LASTQUIESCE`
- `LASTCOPY`
- `LASTARCHQ`
- `LASTSHUTDOWN`
- `X’logPoint ’`
- `FROMLOGPOINT` page 257
- `begin : end`
- `AUX` NO / YES
- `XML`
- `LOB`
- `HISTORY`
- `OBJECTSET` creator.name

**Figure 31** LOGSCAN syntax—Single index specification

### Single index specification

- `INDEX` page 252
- `INDEXSPACE` page 252
- `INDEXES` page 252
- `creatorID`
- `indexName`
- `( )
- `creator.name`
- `INDEX`
- `FROMRBA`
- `FROMLOGPOINT` page 257
- `LASTQUIESCE`
- `LASTCOPY`
- `LASTARCHQ`
- `LASTSHUTDOWN`
- `X’logPoint ’`
Figure 32  LOGSCAN syntax—Multiple index specification

Figure 33  LOGSCAN syntax—Point-in-time recovery specification
LOGSCAN command dependencies and prohibitions

The dependencies and prohibitions for using the LOGSCAN command are as follows:

- A LOGSCAN command cannot be included in the same SYSIN with any of the following statements:
  - RECOVER TABLESPACE
  - RECOVER INDEXSPACE
  - RECOVER INDEX
  - RECOVER UNLOADKEYS
  - RECOVER BUILDINDEX
  - REBUILD INDEX

- If you specify FROMRBA or FROMLOGPOINT, you must specify LOGAPPLY ONLY as well.

- If you specify BACKOUT, you cannot specify LOGONLY or LOGAPPLY ONLY.

- The LOGSCAN command does not consider change accumulation files.

Additionally, consider the following items when you scan indexes:

- If you have used the BMC PACLOG for DB2 utility to exclude the index log records, the LOGSCAN command cannot be used for such indexes.

- If you specify LOGSCAN INDEX (ALL) or LOGSCAN INDEXSPACE (ALL), LOGSCAN INDEX and LOGSCAN INDEXSPACE do not support the FROMRBA/FROMLOGPOINT options. You can use separate index specifications with LOGSCAN INDEX and LOGSCAN INDEXSPACE to use those options.

- If a table space is scanned to a prior point in time, the default TORBA/TOLOGPOINT value for recovery of any indexes on that table space is the log point specified for the table space. If you specify different log points for the table space and its index with TORBA/TOLOGPOINT, RECOVER PLUS ends with an error.

- RECOVER PLUS does not prevent a scan a specific point in time for an index that does not include a scan for the table space or other indexes on the table space.

- Indexes must always have a copy or other backup if LOGSCAN INDEX or LOGSCAN INDEXSPACE is used without BACKOUT. If an index is created on a table that contains data, DB2 does not log the index updates. Similarly, index rebuilds or updates that result from LOG YES utilities are not logged. If the index is created on a table, an image copy or other backup must be made before the index can be scanned.
In addition to the LOG NO SYSCOPY events that make a table space
unrecoverable, any LOAD LOG YES or REORG LOG YES event between the image
copy and the TORBA/TOLOGPOINT value makes an index unrecoverable.
REBUILD INDEX and REORG INDEX can also render an index unrecoverable,
and RECOVER PLUS cannot detect these events if non-BMC utilities are used.

- You cannot use LOGSCAN INDEX or LOGSCAN INDEXSPACE through a point-
in-time recovery event created by using BACKOUT.

LOGSCAN option descriptions

LOGSCAN provides information about the number and size of log records required
for recovery. You can use this information to size the log sort. A report is printed
describing the log records for each object named in the LOGSCAN.

This section describes the options available with LOGSCAN. Not all of the options
described are valid in a single LOGSCAN TABLESPACE, LOGSCAN INDEX, or
LOGSCAN INDEXSPACE command. The recovery strategy that best meets your
needs determines which options you should include in a single command statement.

The LOGSCAN TABLESPACE, LOGSCAN INDEX, and LOGSCAN INDEXSPACE
commands specify one or more table spaces or indexes, or table space or index data
sets, for scanning. A single table space, index, or data set can be scanned to a specific
log point or to the current state (by using all of the log records that apply). Multiple
table spaces, indexes, or data sets can also be scanned to a specific log point or to the
current state. You can have as many LOGSCAN TABLESPACE, LOGSCAN INDEX,
or LOGSCAN INDEXSPACE commands in a job step as necessary; however,
RECOVER PLUS examines all command statements in the job step before processing
starts so that all log processing can be combined and activities can be optimally
scheduled.

Table space specification

The syntax diagram for the Table Space Specification is in Figure 30 on page 245.

TABLESPACE databaseName.tableSpaceName

The TABLESPACE option specifies the table space to be scanned as follows:

- databaseName is the name of the database containing the table space. The default is
  DSNDNB04. The name cannot be DSNDNB01, DSNDNB06, DSNDNB07, or any database
  that is defined with TYPE='W' in SYSIBM.SYSDATABASE.
**tableSpaceName** is the name of the table space within the named database. BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.

**OBJECTSET**

Use the OBJECTSET option with TABLESPACE to process only the table spaces that are in a defined group. For more information, see “OBJECTSET specification” on page 255 and “Using BMC RECOVERY MANAGER groups” on page 476.

---

**NOTE**

To include the indexes in a log scan using TABLESPACE OBJECTSET, specify INDEXES YES (page 249).

---

**INDEXES**

The INDEXES option allows you to specify that you want RECOVER PLUS to process the indexes associated with the table space(s) given by the TABLESPACE option of the LOGSCAN command. The default is INDEXES NO indicating no indexes processed.

---

**NOTE**

The use of INDEX is synonymous to INDEXES for this option.

The INDEXES option is not applicable to INDEXSPACE or INDEX specifications.

---

**INDEXES NO**

Specifying INDEXES NO tells RECOVER PLUS to not process the indexes for the specified table space or table spaces.

**INDEXES YES**

Specifying INDEXES YES tells RECOVER PLUS to process of all indexes associated with the table space(s) specified by the TABLESPACE.

---

**NOTE**

INDEXES YES is invalid with an unqualified OBJECTSET specification (OBJECTSET without TABLESPACE).
LOGSCAN option descriptions

DSNUM

DSNUM specifies a single data set in the specified table space or the entire table space.

DSNUM ALL

DSNUM ALL is the default and specifies that all data sets in the table space are to be scanned. If the table space is nonpartitioned and DSNUM ALL is specified or implied, image copies that are made for specific data sets will be considered only if they are registered at the same log point.

DSNUM integer

Use this option to scan only one partition or data set in the table space. For partitioned table spaces, DSNUM integer specifies the number (from 1 through 4096) of a single partition in the specified table space. For nonpartitioned table spaces, DSNUM integer specifies the number (from 1 to 32) of a single data set of the specified table space.

DSNUM begin : end

Use this option to recover a range of partitions in the table space. For partitioned table spaces, begin specifies the number (from 1 through 4095) of the first partition in the range and end specifies the number (from 2 through 4096) of the last partition in the range. The two numbers are separated by a colon (:) with or without spaces. Wrapping partition numbers (for example, DSNUM 4050 : 300) is not supported.

AUX

The AUX option specifies if auxiliary objects will be included with the scan of the base table spaces. For more information, see “AUX” on page 122.

FROMRBA or FROMLOGPOINT

You can specify FROMRBA or FROMLOGPOINT only with the LOGAPPLY ONLY option. When you use LOGAPPLY ONLY, you must specify FROMRBA or FROMLOGPOINT for each table space and index specification.

FROMRBA or FROMLOGPOINT specifies the log point in the log where a scan of the log records should begin for the object or object data set being scanned. You can specify the log point with a keyword or with a hexadecimal string.

This option assumes that the object or objects have been brought to an appropriate point by some other activity prior to this step.
You can use the FROMRBA and FROMLOGPOINT keywords interchangeably, regardless of the version of DB2 that you are using.

**FROMRBA LASTQUIESCE** or **FROMLOGPOINT LASTQUIESCE**

FROMRBA LASTQUIESCE or FROMLOGPOINT LASTQUIESCE specifies that the log point where a scan of the log records will begin for this object (or object data set) is the log point of the most recent quiesce point registered in SYSIBM.SYSCOPY. If this is recovery of an index, the QUIESCE entry for the table space is used. You can abbreviate LASTQUIESCE to LASTQ.

**NOTE**

If you use LASTQUIESCE and the quiesce is prior to the ALTER ADD PART, the recovery will fail. You must specify a hard coded RBA. If the partition was added for a partition-by-growth universal table space, you can use the LASTQUIESCE option.

**FROMRBA LASTCOPY** or **FROMLOGPOINT LASTCOPY**

FROMRBA LASTCOPY or FROMLOGPOINT LASTCOPY specifies that the log point where a scan of the log records will begin for this object (or object data set) is the log point of the most recent image copy (full or incremental) registered.

**FROMRBA LASTARCHQ** or **FROMLOGPOINT LASTARCHQ**

FROMRBA LASTARCHQ or FROMLOGPOINT LASTARCHQ specifies that the log point of the most recent ARCHIVE LOG MODE(QUIESCE) command is the point where a scan of the log records will begin for this object (or object data set).

**FROMRBA LASTSHUTDOWN** or **FROMLOGPOINT LASTSHUTDOWN**

FROMRBA LASTSHUTDOWN or FROMLOGPOINT LASTSHUTDOWN specifies that the log point of the most recent successful STOP DB2 command is the point where a scan of the log records will begin for this object (or object data set).

**NOTE**

If you specify this keyword in a data sharing environment, log records since the last successful shutdown of the member subsystem on which the job is running are used to scan, without regard to other member subsystems. For this reason, use extreme caution when you use FROMRBA LASTSHUTDOWN or FROMLOGPOINT LASTSHUTDOWN in a data sharing environment to ensure that the shutdown used is really a point consistent with the state of the spaces.
LOGSCAN option descriptions

FROMRBA X'logPoint' or FROMLOGPOINT X'logPoint'

FROMRBA X'logPoint' or FROMLOGPOINT X'logPoint' specifies the log point in the log where a scan of the log records for this object will begin.

'logPoint' is a string of up to twelve hexadecimal digits.

Single index specification

The syntax diagram for the Single Index Specification is in Figure 31 on page 245.

INDEX creatorID.indexName

The INDEX option specifies the index to be scanned as follows:

- **creatorID** is the qualifier creator ID for the index. The default is the user identifier for the utility.

- **indexName** is the name of the index. Indexes for BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.

**OBJECTSET creator.name**

Use the OBJECTSET option with INDEX to process only the indexes that are in a defined group. For more information, see “OBJECTSET specification” on page 255 and “Using BMC RECOVERY MANAGER groups” on page 476.

INDEXSPACE databaseName.indexSpaceName

The INDEXSPACE option specifies the index space to be scanned.

- **databaseName** is the name of the database containing the index space. The default is DSNDB04. The name cannot be DSNDB01, DSNDB06, DSNDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE.

- **indexSpaceName** is the name of the index space within the named database. Indexes for BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.

**DSNUM**

DSNUM specifies either a single data set for the specified index or the entire index.

**DSNUM ALL**

DSNUM ALL is the default and specifies that all of the data sets for the index are to be scanned.
If the index is nonpartitioned and DSNUM ALL is specified or implied, image copies made for specific data set will be considered only if they are registered at the same log point.

For multi-data-set, nonpartitioned index spaces, if the index copy used is a DSN1COPY, DSNUM ALL is not allowed.

**DSNUM integer**

Use this option to scan only one partition or data set for an index. For partitioned indexes, DSNUM integer specifies the number (from 1 through 4096) of a single partition of the specified index. For nonpartitioned indexes, DSNUM integer specifies the number (from 1 to 4096) of a single data set for the specified index.

**DSNUM begin : end**

Use this option to recover a range of partitions in the table space. For partitioned table spaces, begin specifies the number (from 1 through 4095) of the first partition in the range and end specifies the number (from 2 through 4096) of the last partition in the range. The two numbers are separated by a colon (:) with or without spaces. Wrapping partition numbers (for example, DSNUM 4050 : 300) is not supported.

**FROMRBA or FROMLOGPOINT**

For a description of FROMRBA and FROMLOGPOINT and their options, see “FROMRBA or FROMLOGPOINT” on page 250.

**Multiple index specification**

The syntax diagram for the Multiple Index Specification is in Figure 32 on page 246.

**INDEX creatorID.indexName**

The INDEX option specifies the index to be scanned.

- **creatorID** is the qualifier creator ID for the index. The default is the user identifier for the utility.

- **indexName** is the name of the index. Indexes for BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.
**OBJECTSET creator.name**

Use the OBJECTSET option with INDEX to process only the indexes that are in a defined group. For more information, see “OBJECTSET specification” on page 255 and “Using BMC RECOVERY MANAGER groups” on page 476.

**INDEX (ALL) TABLESPACE databaseName.tableSpaceName**

(ALL) TABLESPACE specifies that all indexes for the named table space are to be scanned. The parentheses around ALL are optional.

TABLESPACE specifies the table space from which all indexes are to be scanned as follows:

- **databaseName** is the name of the database containing the table space. The default is DSNDB04. The name cannot be DSNDB01, DSNDB06, DSNDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE.

- **tableSpaceName** is the name of the table space within the named database. BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.

**AUX**

The AUX option specifies if auxiliary objects will be included with the scan of the base table spaces. For more information, see “AUX” on page 122.

**INDEXSPACE databaseName.indexSpaceName**

The INDEXSPACE option specifies indexes to be scanned as follows:

- **databaseName** is the name of the database containing the index space. The default is DSNDB04. The name cannot be DSNDB01, DSNDB06, DSNDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE.

- **indexSpaceName** is the name of the index space within the named database. Indexes for BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.

**INDEXSPACE (ALL) TABLESPACE databaseName.tableSpaceName**

INDEXSPACE (ALL) TABLESPACE specifies that all indexes for the named table space are to be scanned.
TABLESPACE specifies the table space from which all indexes are to be scanned as follows:

- `databaseName` is the name of the database to which the table space belongs. The name cannot be DSNDB01, DSNDB06, DSNDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE. The default is DSNDB04.

- `tableSpaceName` is the name of the table space whose indexes are to be scanned. BMCUTIL, BMCSYNC, and BMCXCOPY (tables that are used by BMC utilities) cannot be used.

If you do not specify the TABLESPACE option, RECOVER PLUS determines the table space for the first valid index specified. All other specified indexes must belong to the same table space.

**CLONE**

The CLONE option indicates that RECOVER PLUS is to scan clone data. When you work with clone data, the following limitations apply:

- You cannot refer to the same object with the CLONE option and without the CLONE option in the same recovery step; you cannot process the base and its clone in the same command.

- Related objects (table space and all indexes for its table, all related LOB objects) should use the same CLONE specification in the same recovery step.

**OBJECTSET specification**

The OBJECTSET option is available in LOGSCAN statement syntax to process logs for groups that are defined using RECOVERY MANAGER. OBJECTSET is available in the following syntactical forms:

- `LOGSCAN OBJECTSET creator.name`

  This form is shown in Figure 29 on page 244. LOGSCAN OBJECTSET performs log scan processing for all objects in a group, both table spaces and indexes.

  **NOTE**

INDEXES YES is not valid with a LOGSCAN OBJECTSET specification.
LOGSCAN option descriptions

- **LOGSCAN TABLESPACE OBJECTSET** `creator.name`

  This form is shown in Figure 30 on page 245. LOGSCAN TABLESPACE OBJECTSET performs log scan processing only for the table spaces in the group.

- **LOGSCAN TABLESPACE OBJECTSET** `creator.name INDEXES YES`

  LOGSCAN TABLESPACE OBJECTSET... INDEXES YES performs log scan processing for the table spaces in the group along with their associated indexes regardless of whether the indexes are included in the group.

- **LOGSCAN INDEX OBJECTSET** `creator.name`

  This form is shown in Figure 31 on page 245. LOGSCAN TABLESPACE OBJECTSET performs log scan processing only for the indexes in the group.

The following rules apply to `creator.name`:

- `creator` specifies the name of the creator of the group and can have up to 128 characters in length.

- `name` specifies the group name and can have up to 128 characters in length.

- You can delimit both `creator` and `name` with single or double quotation marks.

- Both `creator` and `name` can contain the special characters $, #, and @ in any position.

RECOVER PLUS uses dynamic grouping to determine the group contents at the time that you run the RECOVER PLUS job containing the OBJECTSET option. Because RECOVER PLUS reads the objects in the group each time the job is executed, objects may be added or removed from the group.

**NOTE**

RECOVER PLUS does not read RECOVERY MANAGER recovery options for the group. If RECOVERY MANAGER recovery options change, you must run RECOVERY MANAGER to pick up the new options and generate the control cards.

Following are restrictions that apply when you use OBJECTSET in any of its forms on the LOGSCAN statement:

- AUX is ignored with any specification of OBJECTSET.

- You cannot specify DSNUM with OBJECTSET. RECOVER PLUS uses the group definition for the object.

- RECOVER PLUS allows only one OBJECTSET clause, and the statement cannot contain additional TABLESPACE, INDEX, or INDEXSPACE clauses.
Point-in-time recovery specification

The following section describes the Point-in-Time Recovery Specification shown in Figure 33 on page 246.

TORBA or TOLOGPOINT

Use TORBA or TOLOGPOINT to scan a table space or index to a prior point in time identified by a log point. (For more information about using TORBA or TOLOGPOINT with table spaces and indexes, see “LOGSCAN command dependencies and prohibitions” on page 247.)

You can use TORBA or TOLOGPOINT interchangeably, regardless of the version of DB2 that you are using.

TORBA LASTQUIESCE (relativeGenerationNumber) or TOLOGPOINT LASTQUIESCE (relativeGenerationNumber)

Use TORBA LASTQUIESCE or TOLOGPOINT LASTQUIESCE to scan to the most recent quiesce point registered in SYSIBM.SYSCOPY for the table space being scanned. For indexes, the most recent quiesce of the table space is used. RECOVER PLUS scans log records with starting log points less than or equal to the most recent quiesce point, unless BACKOUT is used. For BACKOUT requests, RECOVER PLUS scans log records with starting log points greater than the most recent quiesce point.

Optionally, you can add a relative generation number in parentheses to indicate the quiesce to use. A relative generation number of (0) uses the most recent image copy. For example, TOLOGPOINT LASTQUIESCE (-1) indicates the use of the quiesce before the most recent quiesce.

If you are scanning multiple table spaces in a single LOGSCAN TABLESPACE command statement, all of the table spaces must have a common last quiesce point. However, if you are using multiple LOGSCAN TABLESPACE command statements to scan multiple table spaces, each table space can have a different last quiesce point. The same is true for indexes if you are using LOGSCAN INDEX or LOGSCAN INDEXSPACE.

LASTQUIESCE can be abbreviated as LASTQ.

NOTE

If you use LASTQUIESCE and the quiesce is prior to the ALTER ADD PART, the recovery will fail. You must specify a hard-coded RBA. If the partition was added for a partition-by-growth universal table space, you can use the LASTQUIESCE option.
TORBA LASTCOMMONQ or TOLOGPOINT LASTCOMMONQ

Use TORBA LASTCOMMONQ or TOLOGPOINT LASTCOMMONQ to scan a set of table spaces and indexes to the most recent common quiesce point registered in SYSIBM.SYSCOPY. The scanning of indexes is based on the common quiesce point of the owning table space. RECOVER PLUS scans log records with starting log points less than or equal to the most recent quiesce point common to all table spaces and indexes listed, unless BACKOUT is used. For BACKOUT requests, RECOVER PLUS scans log records with starting log points greater than the most recent common quiesce point. If no common quiesce point exists for all table spaces and indexes specified, an error message is issued.

If you specify this option when only one table space or index is scanned, the effect is exactly the same as using TORBA LASTQUIESCE.

You can use TORBA LASTCOMMONQ or TOLOGPOINT LASTCOMMONQ with LOGSCAN INDEX (ALL) or when multiple index space specifications exist in one command statement.

TORBA LASTARCHQ or TOLOGPOINT LASTARCHQ

Use TORBA LASTARCHQ or TOLOGPOINT LASTARCHQ to scan to the log point of the last ARCHIVE LOG MODE(QUIESCE) command issued for the DB2 subsystem. RECOVER PLUS scans only log records with starting log points less than or equal to the most recent ARCHIVE LOG MODE(QUIESCE) command, unless BACKOUT is used. For BACKOUT requests, RECOVER PLUS scans log records with starting log points greater than the most recent ARCHIVE LOG MODE(QUIESCE) command.

When you use this option, RECOVER PLUS issues a message indicating the date and time of the ARCHIVE LOG MODE(QUIESCE) command used.

TORBA LASTSHUTDOWN or TOLOGPOINT LASTSHUTDOWN

Use TORBA LASTSHUTDOWN or TOLOGPOINT LASTSHUTDOWN to scan to the log point of the last normal system shutdown. RECOVER PLUS scans only log records with starting log points less than or equal to the most recent successful subsystem shutdown, unless BACKOUT is used. For BACKOUT requests, RECOVER PLUS scans log records with starting log points greater than the most recent successful subsystem shutdown.

When you use this option, RECOVER PLUS issues a message indicating the date and time of the last successful system shutdown. You cannot use the LASTSHUTDOWN keyword to scan to the log point of an abnormal DB2 termination. When you use this keyword, ensure that the table spaces and indexes were not in an exception status at the shutdown.
NOTE

In a data sharing environment, this keyword is used to scan to the last successful shutdown of the member subsystem on which the job is running, without regard to other member subsystems. For this reason, use extreme caution when you use TORBA LASTSHUTDOWN or TOLOGPOINT LASTSHUTDOWN in a data sharing environment to ensure that the shutdown used is really a point of data consistency.

To ensure that the last successful shutdown is really a point of data consistency, one of the following conditions must be true:

- The subsystem on which the RECOVER PLUS job is running was the last member of a data sharing group to be stopped.
- All updates to the table spaces and indexes to be scanned occurred on the subsystem on which the RECOVER PLUS job is running.

TORBA X'logPoint' or TOLOGPOINT X'logPoint'

Use TORBA X'logPoint' or TOLOGPOINT X'logPoint' to scan to a prior point in time identified by the log point, 'logPoint'. RECOVER PLUS scans only log records with starting log points less than or equal to 'logPoint', unless BACKOUT is used. For BACKOUT requests, RECOVER PLUS scans log records with starting log points greater than 'logPoint'.

'logPoint' is a string of up to twelve hexadecimal digits.

BACKOUT

BACKOUT invokes the backout strategy for point-in-time recovery by using log points (TORBA, TOLOGPOINT). This strategy assumes that spaces are undamaged and that you require a reset to a specific point in time. The spaces are used with the log records between the point in time and the current point to back out to the required state. Using BACKOUT may enhance the performance of a point-in-time LOGSCAN request significantly.

The following restrictions apply:

- The space must be current as of the last logged activity and not damaged in any way. Multiple-data set, nonpartitioned spaces must have all data sets scanned (DSNUM ALL).
- No LOAD or REORG events can exist between the log point the specified and the current log points. For indexes, no REBUILD INDEX events can exist in this range.
- No prior point-in-time recovery with a START_RBA greater than the log point requested and a PIT_RBA less than the log point requested can exist.

NOTE

In a data sharing environment, this keyword is used to scan to the last successful shutdown of the member subsystem on which the job is running, without regard to other member subsystems. For this reason, use extreme caution when you use TORBA LASTSHUTDOWN or TOLOGPOINT LASTSHUTDOWN in a data sharing environment to ensure that the shutdown used is really a point of data consistency.

To ensure that the last successful shutdown is really a point of data consistency, one of the following conditions must be true:

- The subsystem on which the RECOVER PLUS job is running was the last member of a data sharing group to be stopped.
- All updates to the table spaces and indexes to be scanned occurred on the subsystem on which the RECOVER PLUS job is running.

TORBA X'logPoint' or TOLOGPOINT X'logPoint'

Use TORBA X'logPoint' or TOLOGPOINT X'logPoint' to scan to a prior point in time identified by the log point, 'logPoint'. RECOVER PLUS scans only log records with starting log points less than or equal to 'logPoint', unless BACKOUT is used. For BACKOUT requests, RECOVER PLUS scans log records with starting log points greater than 'logPoint'.

'logPoint' is a string of up to twelve hexadecimal digits.

BACKOUT

BACKOUT invokes the backout strategy for point-in-time recovery by using log points (TORBA, TOLOGPOINT). This strategy assumes that spaces are undamaged and that you require a reset to a specific point in time. The spaces are used with the log records between the point in time and the current point to back out to the required state. Using BACKOUT may enhance the performance of a point-in-time LOGSCAN request significantly.

The following restrictions apply:

- The space must be current as of the last logged activity and not damaged in any way. Multiple-data set, nonpartitioned spaces must have all data sets scanned (DSNUM ALL).
- No LOAD or REORG events can exist between the log point the specified and the current log points. For indexes, no REBUILD INDEX events can exist in this range.
- No prior point-in-time recovery with a START_RBA greater than the log point requested and a PIT_RBA less than the log point requested can exist.
BACKOUT may not be requested with LOGONLY or LOGAPPLY ONLY.

**NOTE**
BACKOUT uses only logs and spaces but requires that the spaces be current. The LOGONLY and LOGAPPLY ONLY options imply scanning log records going forward by using a space restored to a previous state.

Change accumulation files are not allowed with BACKOUT because they are not properly ordered. Output accumulation files are also not supported because they are defined from the point of the last image copy to the current point.

**LOGAPPLY ONLY**

LOGAPPLY ONLY specifies that only log information is to be scanned for a table space or index and that no image copy data sets are to be used. Use this option with FROMRBA or FROMLOGPOINT.

When you want to scan only log records for multiple table spaces or indexes in a single LOGSCAN TABLESPACE, LOGSCAN INDEX, or LOGSCAN INDEXSPACE command, you need to specify only a single LOGAPPLY ONLY. However, you must specify FROMRBA or FROMLOGPOINT for each table space or index.

LOGAPPLY ONLY is not valid with BACKOUT.

**LOGONLY**

LOGONLY scans only the log records to the data sets. All log records written after a point recorded in the data set are used. Specify LOGONLY when the data sets of the target objects have been restored by using another process and the HPGRBRBA value in the header pages is correctly set to indicate where log apply should begin. You can use this option on single table spaces or indexes, or on table space or index lists.

LOGONLY is not valid with BACKOUT.
Using multiple commands

Multiple RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE, REBUILD INDEX, and RECOVER UNLOADKEYS commands are allowed in a single job. However, because RECOVER PLUS attempts to coordinate all requested work, certain restrictions apply. Only a single RECOVER BUILDINDEX command is allowed and, when present, must be the only command for the job step. The RECOVER BUILDINDEX command supports only one table space but does support multiple indexes on that table space.

**NOTE**

RECOVER PLUS does not support commands in the input data set that execute other utilities.

General restrictions for multiple commands

The following general restrictions apply when you use multiple command statements:

- No space may be named or implied for recovery in a RECOVER PLUS job more than once. For the purposes of this restriction, the data sets of a table space are considered separate objects. For example, the following command statements would be incorrect if AFRIND1 was an index on a table in table space AFRDB.AFRTS:

```sql
REBUILD INDEX (AFRIND1)
REBUILD INDEX (ALL) TABLESPACE AFRDB.AFRTS
```

However, the following command statements are correct:

```sql
REBUILD INDEX (AFRIND1) PART 1
REBUILD INDEX (AFRIND1) PART 2
REBUILD INDEX (AFRIND2)
```

If RECOVER UNLOADKEYS command statements are present for any partition of a table space, REBUILD INDEX command statements on nonpartitioned indexes for the same table space are not valid. However, REBUILD INDEX command statements for the partitioned index on the space are valid.
For example, the following command statements are incorrect if AFRIND2 is a nonpartitioned index:

```
REBUILD INDEX (AFRIND2) TABLESPACE AFRDB.AFRTS
RECOVER UNLOADKEYS (AFRIND3) PART 1 TABLESPACE AFRDB.AFRTS
```

Because the first command statement requires that all partitions of the table space be read to extract keys for the nonpartitioned index, you should perform one of the following actions:

— unload keys for both indexes in one or more jobs and follow these jobs with a RECOVER BUILDINDEX job

— recover both indexes in one job by using REBUILD INDEX

The following command statements are correct:

```
REBUILD INDEX (AFRIND1) PART 1
RECOVER UNLOADKEYS (AFRIND2, AFRIND3) PART 1
```

This execution rebuilds the partitioned index while extracting the keys for nonpartitioned indexes from partition 1 of the space. RECOVER PLUS processes these two command statements concurrently.

- RECOVER BUILDINDEX, when present, must be the only command in the job step.

- The ANALYZE option may be on any command statement and overrides the default (ANALYZE YES). If you specify the option on more than one RECOVER PLUS command statement, the value (YES, NO, or ONLY) must be the same in all cases.

- When you are recovering a cloned object, all related objects named in the job must also be clones. For example, the following command statements are incorrect:

```
RECOVER TABLESPACE AFRDB.AFRTS CLONE
RECOVER INDEX(ALL) TABLESPACE AFRDB.AFRTS
```
Restrictions for STOGROUP-defined objects

When you want to recover multiple STOGROUP-defined objects with some objects reallocated and others not reallocated (by using the REDEFINE option), you must use one command statement to reallocate objects and a second command statement for those objects that are not reallocated, as follows:

- For table space recovery, use one RECOVER TABLESPACE command statement for table spaces that are reallocated and a second RECOVER TABLESPACE command statement for table spaces that are not reallocated. For more information, see “REDEFINE” on page 191.

- To rebuild an index space by using REBUILD INDEX, use one command statement for index spaces that are reallocated and a second command statement for index spaces that are not reallocated. For more information, see “REDEFINE” on page 226.

- For index space recovery by using RECOVER BUILDINDEX, use one command statement for index spaces that are reallocated and a second command statement (in a separate execution) for index spaces that are not reallocated. For more information, see “REDEFINE” on page 242.

General information about SORTDEVT and SORTNUM

You can specify SORTDEVT and SORTNUM in the installation options module, or with the following RECOVER PLUS commands:

- OPTIONS
- RECOVER TABLESPACE, RECOVER INDEXSPACE, or RECOVER INDEX
- RECOVER UNLOADKEYS
- REBUILD INDEX

SORTDEVT and SORTNUM values override other values as follows:

- values on the preceding RECOVER commands or the REBUILD INDEX command override the values on the OPTIONS command

- values on the OPTIONS command override the values in the installation options module

If you specify SORTNUM with the OPTIONS, RECOVER, or REBUILD commands, and you do not specify SORTDEVT with any of the commands or you define SORTDEVT in the installation options module, RECOVER PLUS ignores SORTNUM.
Restrictions for key work data sets, SKEYDDN, SORTNUM, and SORTDEVT

You may have multiple REBUILD INDEX command statements, RECOVER UNLOADKEYS command statements, or both. Restrictions on the key work data sets and the SORTNUM and SORTDEVT options of these statements are designed to allow the most efficient processing of all command statements. The restrictions are determined by whether the table space for the indexes is partitioned or nonpartitioned and, if partitioned, whether nonpartitioned indexes are being rebuilt or whether RECOVER UNLOADKEYS is specified.

Restriction for nonpartitioned table space recoveries

You do not need to specify multiple REBUILD INDEX command statements for one nonpartitioned table space, but with RECOVER PLUS you can specify multiple statements. However, because RECOVER PLUS unloads the keys concurrently for all of the indexes, the key work data sets and the SORTDEVT and SORTNUM options must be the same on all statements for one table space.

Restrictions for partitioned table spaces

Restrictions for partitioned table spaces vary according to whether you are rebuilding a partitioned or nonpartitioned index and whether you specify RECOVER UNLOADKEYS.

Partitioned index rebuild

If you specify only partitioned index recovery, no restrictions exist on the key work data sets or the SORTNUM and SORTDEVT options. The following examples are allowed:

```
REBUILD INDEX (AFRIND1) PART 1 WORKDDN KEYS
REBUILD INDEX (AFRIND1) PART 2 NOWORKDDN
REBUILD INDEX (AFRIND1) PART 1 WORKDDN KEYS
REBUILD INDEX (AFRIND1) PART 2 WORKDDN KEYS
```

**NOTE**

You can use WORKDDN only with MAXKSORT 1.
Nonpartitioned index rebuild

When you specify REBUILD INDEX for a nonpartitioned index, the key work data sets and the SORTDEVT and SORTNUM options must agree on all REBUILD INDEX command statements for the table space, including those for all of a partitioned index or parts of a partitioned index. This agreement allows RECOVER PLUS to extract keys for all indexes from each partition during one pass of the table space.

RECOVER UNLOADKEYS specified

When you specify RECOVER UNLOADKEYS, a data set is created to hold the keys for a subsequent job that builds the index. The keys for all partitions of a table space within the step are required to be in one data set that is designated by the SKEYDDN ddname. Generally, if you run RECOVER UNLOADKEYS jobs for every table space partition, you can code one of the following command statements (in each job):

```
RECOVER UNLOADKEYS(ALL) TABLESPACE...PART nn
RECOVER UNLOADKEYS (indexName1,...,indexNamen) PART nn
```

If you specify multiple RECOVER UNLOADKEYS command statements for the table space, you must use the same SKEYDDN option on these command statements. If other REBUILD INDEX or RECOVER UNLOADKEYS command statements reference other table spaces or other partitions, the keys for different table spaces must be unloaded to different data sets. Name these data sets with the SKEYDDN option.

For example, you could code the following command statements:

```
RECOVER TABLESPACE AFRDB.AFRTS1 DSNUM 1
   TABLESPACE AFRDB.AFRTS2 DSNUM 1
RECOVER UNLOADKEYS (ALL) TABLESPACE AFRDB.AFRTS1 PART 1
   REBUILD INDEX (AFRIND1) TABLESPACE AFRDB.AFRTS1 PART 1
RECOVER UNLOADKEYS (ALL) TABLESPACE AFRDB.AFRTS2 PART 1
   SKEYDDN TS2K
   REBUILD INDEX (AFRCLUS) TABLESPACE AFRDB.AFRTS2 PART 1
```

The keys for the nonpartitioned indexes of AFRDB.AFRTS1 are directed to the default DDNAME SKEY. The keys for the nonpartitioned indexes of AFRDB.AFRTS2 are directed to the DDNAME TS2K. SKEY and TS2K are ddnames, not prefixes. The keys for the partitioned indexes AFRIND1 and AFRCLUS are directed to NOWORKDDNN by default.
Restrictions for key work data sets, SKEYDDN, SORTNUM, and SORTDEVT
# Building and executing RECOVER PLUS jobs

This chapter explains how to build a job and how to run the utility, including starting, restarting, and terminating a RECOVER PLUS job.

## Overview

- Building RECOVER PLUS jobs
  - JOB statement
  - EXEC statement
  - REGION parameter
  - Utility parameters on the EXEC statement
  - STEPLIB DD or JOBLIB statements
  - RECOVER PLUS data sets and RECOVER PLUS DD statements
  - Defining sort work data sets
  - Determining sort work space requirements
  - Determining unloaded keys file space requirements
  - Specifying RECOVER PLUS syntax

- Running RECOVER PLUS jobs
  - Starting a RECOVER PLUS job
  - Restarting a RECOVER PLUS job
  - Displaying the status of a RECOVER PLUS job
  - Terminating a RECOVER PLUS job
  - Cleaning up the BMCUTIL and BMCSYNC tables
  - Cleaning up after a RECOVER UNLOADKEYS job

- Using MSGLEVEL to control RECOVER PLUS output
  - Sample output with MSGLEVEL(0)
  - Sample output with MSGLEVEL(1)
  - Sample output with MSGLEVEL(2)

- Examples of RECOVER PLUS operations
  - Multiple table space recovery and index rebuilds
  - Parallel index key sorts and multitasking index rebuilds
  - Multiple log sorts, MERGE phases, and parallel index key sorts with multitasking index rebuilds
  - Multiple table space and index recoveries
  - BACKOUT recovery
  - Instant Snapshot recovery
Overview

This chapter explains how to build a job and describes all of the utility parameters. Instructions about how to perform the following tasks are also included:

- starting a RECOVER PLUS job
- restarting a job
- terminating a job

Building RECOVER PLUS jobs

Building a job for the RECOVER PLUS product involves creating a set of JCL that includes:

- a JOB statement (page 269)
- an EXEC statement with the appropriate utility parameters (page 269 and page 270)
- STEPLIB or JOBLIB DD statements as needed (page 276)
- DD statements as needed for the data sets needed for the recovery (page 276 to page 290)
- RECOVER PLUS statements using the appropriate command syntax (page 291)

A simple example of a RECOVER PLUS job is illustrated in Figure 34. For more examples of RECOVER PLUS jobs, see Chapter 5, “Examples of RECOVER PLUS jobs.”

Figure 34  JCL for a simple RECOVER PLUS execution

```
//DMBRECV2 JOB (5212), 'SIMPLE2', CLASS=A,
// MSGCLASS=X, NOTIFY=AFR
//BMCRVCVR EXEC PGM=AFRMAIN, REGION=0M,
// PARM='DB21,ROADMBSM2X2,NEW,MSGLEVEL(1)'
//STEPLIB DD DISP=SHR, DSN=product.libraries
// DD DISP=SHR, DSN=DB2.DSNEXIT
// DD DISP=SHR, DSN=DB2.DSNLOAD
//SYSIN DD *
RECOVER TABLESPACE AFRDB20.AFRTS21
REBUILD INDEX (AFR.AFRIX21) NOWORKDDN
```
JOB statement

Include a RECOVER PLUS JOB statement that conforms to your site’s standards. You can include the REGION parameter on either your JOB statement or your EXEC statement. See “REGION parameter” for recommendations.

EXEC statement

The RECOVER PLUS EXEC statement specifies PGM =AFRMAIN. The EXEC statement also specifies RECOVER PLUS utility parameters, which are described in “Utility parameters on the EXEC statement” on page 270.

You can include the REGION parameter on either your EXEC statement or your JOB statement. See “REGION parameter” for recommendations.

REGION parameter

Include the REGION parameter on either your JOB statement or your EXEC statement to specify the region size (the amount of virtual storage used by the utility). For the best performance, BMC recommends that you specify REGION=0M to allocate all available virtual storage to the RECOVER PLUS job. If your data center does not permit you to specify REGION=0M, specify 8 MB to ensure adequate storage. A typical RECOVER PLUS step requires between 5 MB and 8 MB of virtual storage for code, control blocks, and I/O buffers.

NOTE

If you are running under DB2 Version 8 and you specify a value for REGION other than 0M, ensure that you have an appropriate value set for the MEMLIMIT parameter, either as your site’s default SMF option or on your JOB statement or EXEC statement.

BMC recommends that you have a MEMLIMIT value of at least 1 GB. For more information, see “Setting the MEMLIMIT parameter” on page 79.

For a discussion of the virtual memory usage when many log files are read concurrently, see “Does LOGSORT have other advantages or disadvantages?” on page 419.
Utility parameters on the EXEC statement

The RECOVER PLUS EXEC statement includes the following utility parameters:

- DB2 subsystem ID or group attach name for data sharing (ssid)
- utility ID (utilID)
- restart parameter (restartParm)
- message level parameter (msgLevel)
- checkpoint override parameter (checkPt)
- parameter for restoring the initial status of DB2 objects (rdb2Stat)
- installation options module for RECOVER PLUS (afrOpts)

The following illustration shows the format of the EXEC statement:

```
//stepName EXEC PGM=AFRMAIN,REGION=0M,  
//         PARM='ssid,utilID,restartParm,msgLevel,checkPt,rdb2Stat,afrOpts'
```

The parameters ssid, utilID, and restartParm are positional and must be specified in the order shown.

You must specify the DB2 subsystem ID or group attach name for DB2 data sharing. However, if you do not specify the utility ID or the restart parameter and want to use the defaults for these parameters, you must include a comma as a place holder for each parameter. For example, in a statement with the following format, the subsystem ID or group attach name (ssid) and the restart parameter (restartParm) are specified and the default is used for the utility ID:

```
//         PARM='ssid,,restartParm'
```

After you specify the DB2 subsystem or group attach name, the utility ID, and the restart parameter, you can specify the remaining parameters in any order; they are non-positional. If you do not specify one of the non-positional parameters, RECOVER PLUS uses the default value for the utility parameter.
**DB2 subsystem identifier (ssid)**

The DB2 subsystem identifier is the one-character to four-character DB2 subsystem ID that identifies where the objects to be recovered reside. If the DB2 subsystem identifier is not coded in the runtime parameters on the JCL EXEC statement for the utility, the default value from the DSNHDECP module that is in the STEPLIB concatenation will be used. (The DSNHDECP module was created when DB2 was installed. The installation-generated DSNHDECP module typically resides in the SDSNEXIT library.)

You can also use the group attachment name (for DB2 data sharing) in place of the ssid parameter. Using the group attachment name, you can use the same JCL but run on any member of a data sharing group. When a job is submitted with a group name, a subsystem in the group must be executing on the system where the job is submitted.

**Utility Identifier (utilID)**

This parameter specifies the ID that uniquely names a utility execution or job step. If you do not specify this parameter, RECOVER PLUS uses the default, userid.jobName. The rules for utility ID are as follows:

- The utility ID can be 1 to 16 characters.
- The utility ID consists of alphanumeric characters, plus the following characters: #, $, @, !, ¬, . , and ¢.

When you run multiple RECOVER PLUS jobs concurrently, each job must use a unique utility ID.

**Restart parameter (restartParm)**

The restart parameter can have one of the values described in this section. The value that you choose for this parameter determines which of the following execution types is invoked:

- a new RECOVER PLUS execution
- a RECOVER PLUS execution that you want to restart
- a run to print control section information to track maintenance that has been applied without performing any other processing

For more information about resubmitting a failed RECOVER PLUS job, see “Restarting a RECOVER PLUS job” on page 292.
Utility parameters on the EXEC statement

**Blank or not specified**

This value is the default. If the utility ID is not present in the BMCUTIL table, this value starts a new RECOVER PLUS job. If the utility ID is already present in the BMCUTIL table, RECOVER PLUS terminates with an error.

**MAINT**

MAINT limits processing to listing the installation options, fixes applied to RECOVER PLUS, and the names of BMC tables. No recovery processing is performed.

---

**NOTE**

Even if the connection to DB2 fails, RECOVER PLUS still prints the maintenance information.

**NEW**

NEW starts a new RECOVER PLUS utility execution. You can reuse a utility ID that already exists in the BMCUTIL and BMCSYNC tables. (These tables are described in Appendix C, “BMC utilities database.”) When you use NEW, any existing utility job with the same ID is replaced, and a new utility job is started. RECOVER PLUS will not allow the job to start if another job with the same utility ID on the same DB2 subsystem or data sharing group is active.

**NEW/RESTART**

If the utility ID already exists in the BMCUTIL table, NEW/RESTART restarts the utility job from the last recorded sync point or (if the last checkpoint taken was not for a sync point) from the beginning of the last incomplete phase. If the utility ID does not already exist, NEW/RESTART initiates a new RECOVER PLUS utility job. This option allows a job to be submitted again without changing the JCL in most cases. Some cases may require a change to DD statements with a disposition of NEW. For more information about sync points and checkpoints, see “Restarting a RECOVER PLUS job” on page 292.

**NEW/RESTART(PHASE)**

NEW/RESTART(PHASE) restarts the utility at the beginning of the last incomplete RECOVER PLUS phase if the utility ID already exists in the BMCUTIL table. Otherwise, NEW/RESTART(PHASE) starts the job as a new RECOVER PLUS utility job. This option allows a job to be submitted again without changing the JCL in most cases. Some cases may require a change to DD statements with a disposition of NEW.
**RESTART**

If the utility ID already exists, RESTART restarts the utility from the last sync point or (if the last checkpoint taken was not for a sync point) from the beginning of the last incomplete phase. If the utility ID does not already exist in the BMCUTIL table, RECOVER PLUS terminates and issues an error message. For more information about sync points, see “Restarting a RECOVER PLUS job” on page 292.

**RESTART(PHASE)**

RESTART(PHASE) restarts the utility at the beginning of the last incomplete phase. A row for the utility ID must exist in the BMCUTIL table or the job terminates with an error message.

**TERM**

Specifying TERM terminates a stopped or failed utility by removing all sync point and restart information for the utility ID from the BMCUTIL and BMCSYNC tables. You can use the TERM parameter to “clean up” after an error condition; however, you may prefer to restart the recovery. RECOVER PLUS does not issue an error message when TERM is specified, and no row exists for the utility ID in the BMCUTIL table.

---

**NOTE**

For additional instructions about “clean up” after use of RECOVER UNLOADKEYS, see “Cleaning up RECOVER UNLOADKEYS entries” on page 578.

**Message level parameter (msgLevel)**

This utility parameter determines which output files and messages RECOVER PLUS returns. Valid values for MSGLEVEL are 0, 1, and 2. If you do not specify msgLevel, RECOVER PLUS uses the default, MSGLEVEL(1). For a description of the output files that are returned by different values of MSGLEVEL, see Table 10 on page 277. For examples of the different output files and their content, see “Using MSGLEVEL to control RECOVER PLUS output” on page 298.

**Checkpoint override parameter (checkPt)**

This utility parameter overrides the CHECKPT installation option. CHECKPT provides a means of controlling the overhead associated with taking checkpoints. Taking a checkpoint refers to the process of recording the end of a phase in the BMCUTIL and BMCSYNC tables. When a checkpoint is taken, message BMC40091 is issued. If you do not specify the checkpoint override parameter, the default is the value of the CHECKPT installation option.
For more information about CHECKPT, see “Checkpoints for RECOVER PLUS restart” on page 91 and Appendix A, “RECOVER PLUS installation options.”

The checkpoint override parameter can have a value of CHECKPT(NO) or CHECKPT(PHASE).

**CHECKPT(NO)**

CHECKPT(NO) causes no checkpoints to be taken, except those necessary to synchronize RECOVER PLUS execution with the execution of other BMC utilities and MERGE checkpoints that are necessary to guarantee the integrity of output copy, point-in-time recovery, or index rebuild registrations. CHECKPT(NO) is recommended for short RECOVER PLUS jobs in which you do not want to incur checkpoint overhead or that you do not mind rerunning entirely if necessary.

**CHECKPT(PHASE)**

CHECKPT(PHASE) causes a checkpoint to be taken at the end of each processing phase if more than the number of minutes specified by the value of the CHECKINT installation option have passed since the last checkpoint was taken. For information about the CHECKINT installation option, see Appendix A, “RECOVER PLUS installation options.”

Specify CHECKPT(PHASE) for longer running jobs when it would be costly to rerun the entire job.

---

**NOTE**

Specifying registered output copies in a RECOVER TABLESPACE, RECOVER INDEXSPACE, or RECOVER INDEX command causes a checkpoint to be taken at the end of the MERGE phase for the table space, index space, or index, even if the required number of minutes has not elapsed. This action protects the update to SYSIBM.SYSCOPY, which registers the copy. No sync points are taken. Checkpoints may be forced to guarantee registration of a point-in-time recovery or a rebuild of an index.

---

**RDB2STAT override parameter (rdb2Stat)**

This utility parameter overrides the RDB2STAT installation option. RDB2STAT indicates the status to which DB2 objects should be returned at the successful completion of a RECOVER PLUS job. If you do not specify the RDB2STAT override parameter, RECOVER PLUS defaults to the setting of the RDB2STAT installation option. For information about the RDB2STAT installation option, see Appendix A, “RECOVER PLUS installation options.”

The RDB2STAT override parameter can have a value of RDB2STAT(YES), RDB2STAT(NO), RDB2STAT(RW), or RDB2STAT(RO).
RDB2STAT(YES)

RDB2STAT(YES) causes RECOVER PLUS to issue DB2 commands to restore the DB2 status of each space after RECOVER PLUS completes the requested operations. For examples of how RECOVER PLUS restores the initial status of a space, see “Restoring initial status” on page 86.

RDB2STAT(NO)

RDB2STAT(NO) causes RECOVER PLUS not to issue DB2 commands to restore the DB2 status of spaces after a RECOVER PLUS run is complete. The spaces remain stopped.

RDB2STAT(RW)

RDB2STAT(RW) causes RECOVER PLUS to issue DB2 commands to set the status of each space to read-write (RW) mode (regardless of the initial status of the space) after the requested operations have completed successfully.

RDB2STAT(RO)

RDB2STAT(RO) causes RECOVER PLUS to issue DB2 commands to set the status of each space to read-only (RO) mode (regardless of the initial status of the space) after the requested operations have completed successfully.

Installation options module parameter for RECOVER PLUS (afrOpts)

The installation options module parameter overrides the installation options module, AFR$OPTS, for RECOVER PLUS. Using this feature, you can provide a different options module at runtime. The length of the parameter determines how the parameter is used:

- If this parameter is greater than four characters in length, the options module specified by the parameter is loaded.

- If this parameter is less than or equal to four characters in length, the parameter is considered a suffix override for the OPTS part of the installation options module name. For example, specifying AFROPTS(TEMP) results in an attempt to load AFR$TEMP as the RECOVER PLUS options module.

NOTE

If RECOVER UNLOADKEYS was specified for an index, the associated table space is placed in RO status.
As an alternative to afrOpt, you can also specify opts on the EXEC statement. For example, either of the following examples is valid:

```
PARM='DFC2,F312396B,NEW,MSGLEVEL(1),AFROPTS(AFR$DB2A)'

PARM='DFC2,F312396B,NEW,MSGLEVEL(1),OPTS(AFR$DB2A)'
```

For information about the AFR$OPTS installation options module, see Appendix A, “RECOVER PLUS installation options.”

### STEPLIB DD or JOBLIB statements

The RECOVER PLUS STEPLIB DD statement must specify the following libraries, unless they are included in your system's LINKLIST or in a JOBLIB statement:

- load libraries that contain the files (including the options modules) for the following BMC products and components:
  - RECOVER PLUS
  - BMCSORT (AUP)
  - DB2 Solution Common Code (SCC)

- libraries that contain any EDITPROCS, VALIDPROCS, FIELDPROCS, and user-written routines

  EDITPROCs are required only when a REBUILD INDEX or RECOVER UNLOADKEYS command is requested for an index on a table with an EDITPROC specified.

- DB2 load library

All load libraries in the STEPLIB or JOBLIB concatenation must be APF authorized. (For more information, see “APF authority” on page 82.)

### RECOVER PLUS data sets and RECOVER PLUS DD statements

Your recovery needs and the RECOVER PLUS strategy that you use determine the data sets needed for recovery. Some data sets are required; others are optional. Some data sets require DD statements in the JCL; others can be dynamically allocated by RECOVER PLUS.
DD statements common to all RECOVER PLUS executions

You can control the amount of output that you receive by specifying a value for the MSGLEVEL utility parameter. Valid values for MSGLEVEL are 0, 1, and 2. If you do not specify MSGLEVEL, RECOVER PLUS uses the default, MSGLEVEL(1). For more information, see “Using MSGLEVEL to control RECOVER PLUS output” on page 298.

Table 10 provides a list of RECOVER PLUS data sets along with specific information about each data set, including

- description of the data set
- whether the data set is required or optional
- whether a specific MSGLEVEL is required to produce the data set
- whether the data set can be dynamically allocated or whether it requires a DD statement

Table 10  RECOVER PLUS data sets (part 1 of 3)

<table>
<thead>
<tr>
<th>Data set name</th>
<th>Required or optional</th>
<th>Dynamically allocated (yes or no)</th>
<th>MSGLEVEL value required&lt;sup&gt;a,b&lt;/sup&gt;</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSIN</td>
<td>required</td>
<td>no</td>
<td>not applicable</td>
<td>defines the input data set that contains the RECOVER PLUS commands and their options&lt;sup&gt;c&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>This data set must be fixed blocked (RECFM=FB) with a record length of 80 (LRECL=80). Columns 73-80 are reserved for sequence numbers; any characters in these columns are ignored by RECOVER PLUS. For details of the RECOVER PLUS commands and options that you can specify, see Chapter 3, “RECOVER PLUS syntax.”</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Unicode is not supported in the SYSIN file.&lt;sup&gt;d&lt;/sup&gt;</td>
</tr>
<tr>
<td>AFRPRINT</td>
<td>required</td>
<td>yes</td>
<td>0, 1, or 2</td>
<td>provides execution message output</td>
</tr>
<tr>
<td>AFRSUMRY</td>
<td>required</td>
<td>yes</td>
<td>0, 1, or 2</td>
<td>lists maintenance applied, phases completed, utility return codes</td>
</tr>
<tr>
<td>AFRSTMT</td>
<td>required</td>
<td>yes</td>
<td>0, 1, or 2</td>
<td>lists input statements, commands, and options as specified in SYSIN, installation option values, log file resources, and messages generated by analyze, planning, and termination stages</td>
</tr>
<tr>
<td>AFROSUM&lt;sup&gt;e&lt;/sup&gt;</td>
<td>optional, based on MSGLEVEL</td>
<td>yes</td>
<td>1 or 2</td>
<td>lists all of the objects being recovered or rebuilt and their status, options, and resources</td>
</tr>
<tr>
<td>Data set name</td>
<td>Required or optional</td>
<td>Dynamically allocated (yes or no)</td>
<td>MSGLEVEL value required(^{a,b})</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>AFRPLAN(^{e})</td>
<td>optional, based on MSGLEVEL</td>
<td>yes</td>
<td>2</td>
<td>records the execution plan</td>
</tr>
<tr>
<td>AFRERR</td>
<td>not applicable</td>
<td>yes</td>
<td>not applicable</td>
<td>records warnings or error messages</td>
</tr>
<tr>
<td>AFRTIME</td>
<td>optional</td>
<td>yes, if needed</td>
<td>not applicable</td>
<td>available only with the Recovery Management for DB2 solution for the recovery estimation feature, reports the ten table spaces for which the longest elapsed time was recorded to recover each table space and all of its indexes</td>
</tr>
<tr>
<td>AFRTRACE</td>
<td>not applicable</td>
<td>yes</td>
<td>not applicable</td>
<td>provides trace information when errors occur during execution</td>
</tr>
<tr>
<td>SYSERR or AFRDUMP</td>
<td>not applicable</td>
<td>yes</td>
<td>not applicable</td>
<td>captures snap dumps issued during error processing If SYSERR is not coded in the JCL, RECOVER PLUS looks for AFRDUMP. If DD names are not coded for either SYSERR or AFRDUMP, RECOVER PLUS attempts to dynamically allocate SYSERR. BMC does not recommend the suppression of dump output because it is used as a troubleshooting aid.</td>
</tr>
<tr>
<td>SYSOUT(^{nnn})</td>
<td>required</td>
<td>yes</td>
<td>not applicable</td>
<td>used for BMCSORT message output</td>
</tr>
<tr>
<td>SYSUDUMP</td>
<td>not applicable</td>
<td>yes</td>
<td>not applicable</td>
<td>captures system abend dumps If SYSUDUMP is not in the JCL, it is dynamically allocated to SYSOUT. BMC recommends the use of SYSUDUMP as a troubleshooting aid.</td>
</tr>
<tr>
<td>SYSSCAN</td>
<td>optional</td>
<td>no</td>
<td>not applicable</td>
<td>contains the output from the LOGSCAN command If the SYSSCAN DD is not specified, the LOGSCAN output is in AFRPRINT.</td>
</tr>
</tbody>
</table>
### Table 10  RECOVER PLUS data sets (part 3 of 3)

<table>
<thead>
<tr>
<th>Data set name</th>
<th>Required or optional</th>
<th>Dynamically allocated (yes or no)</th>
<th>MSGLEVEL value required&lt;sup&gt;a,b&lt;/sup&gt;</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFRGDG</td>
<td>optional</td>
<td>no</td>
<td>not applicable</td>
<td>used with dynamic allocation to provide a GDG base if one does not exist. This data set contains the control cards needed to perform an IDCAMS DEFINE operation and the symbolic variable &amp;BASE. For more information about the use of GDGs, see “GDGs and symbolic variables in data set name construction” on page 454.</td>
</tr>
<tr>
<td>SYSPICK</td>
<td>optional</td>
<td>no</td>
<td>not applicable</td>
<td>lists all of the input tape and cartridge volumes that are allocated during recovery, including those implicated because of a need to recall archived data sets to disk prior to recovery. SYSPICK output consists of two reports. Both reports contain volume serial numbers, device types, data set names, and sequence numbers. One report orders the data by volume serial number and the other report orders data based on the order in which RECOVER PLUS uses the data sets.</td>
</tr>
<tr>
<td>AFRDBG</td>
<td>optional</td>
<td>no</td>
<td>not applicable</td>
<td>used to help with performance and problem diagnosis associated with Instant Snapshots, if you have EXTENDED BUFFER MANAGER (XBM) version 5.6 or later and you use Instant Snapshots for recovery.</td>
</tr>
</tbody>
</table>

<sup>a</sup> The values shown are the required values of MSGLEVEL to produce the output file. The default value of MSGLEVEL is 1.

<sup>b</sup> When “not applicable” is shown, RECOVER PLUS produces the data set regardless of the value of MSGLEVEL. In some cases the files are dynamically allocated and in other cases they are not.

<sup>c</sup> For descriptions of the RECOVER PLUS commands and options, see Chapter 3, “RECOVER PLUS syntax.”. Sections are included with information on the use of Long Names and Unicode.

<sup>d</sup> However, RECOVER PLUS can process spaces that contain tables, indexes, or storage groups with Unicode names. (Spaces cannot contain Unicode names). RECOVER PLUS commands that use wild cards do not include objects that match the pattern but contain Unicode characters that are not translatable to EBCDIC in the wild card position. This is because SYSIN is EBCDIC and wild card processing is done in EBCDIC.

<sup>e</sup> RECOVER PLUS produces these data sets regardless of the value of MSGLEVEL when you specify OPTIONS ANALYZE ONLY.
Data sets and DD statements for RECOVER TABLESPACE, RECOVER INDEXSPACE, or RECOVER INDEX

The options specified, or for which defaults are accepted, in a RECOVER TABLESPACE, RECOVER INDEXSPACE, or RECOVER INDEX command statement may require that you specify DD statements in the JCL in addition to those common to all RECOVER PLUS executions (see “RECOVER PLUS data sets and RECOVER PLUS DD statements” on page 276). Use DD statements to specify the following data sets:

- any requested output image copy data sets that are not dynamically allocated
- log sort work data sets

For examples that use dynamic allocation and DD statement construction for output copy DD statements, see Chapter 5, “Examples of RECOVER PLUS jobs.”

Input image copy data sets

Normally dynamically allocated, input image copy data sets are needed for recovery unless recovery is possible by using only the log (for example, with LOGAPPLY ONLY or through a point-in-time recovery with BACKOUT).

DD statements are allowed, but BMC does not recommend their use for input image copy data sets. Use a DD statement for an input copy only to override a dynamic allocation. If you use a DD statement, it must contain a unit and a VOL=SER for uncataloged data sets. RECOVER PLUS matches the information in the DD statement to the data set that RECOVER PLUS would otherwise dynamically allocate by data set name and VOL=SER if uncataloged. Any DDNAME can be specified.

You do not need to code DD statements to control tape mounts for stacked tapes. RECOVER PLUS automatically selects the proper order and handling to accommodate this.

Output image copy data sets

Output image copy data sets are required only when you are requesting one or more output copies. Either ddnames or ddname prefixes are given in the syntax (or defaults are used), and these names or prefixes correspond to the JCL. Another option is to use dynamic allocation of output image copy data sets (see “Allocating output image copy data sets dynamically” on page 452).

NOTE

Dynamic allocation of output copies is required for spaces with more than 254 partitions.
The ddnames are constructed differently for partitioned and nonpartitioned spaces:

- **DDname Construction for Partitioned Spaces**

  If you have coded OPTIONS OUTCOPY BYPART, or if the OUTCOPY installation option is set to BYPART, and you have coded DSNUM on the RECOVER TABLESPACE, RECOVER INDEXSPACE, or RECOVER INDEX command statement, you specify the ddname as follows:

  \[ DDName{nn} \text{ or } DDName{nnn} \]

  Where \( DDName \) is the prefix that you specified, and \( nn \) (for spaces with fewer than 100 partitions) or \( nnn \) (for spaces with 100 or more partitions) is the partition number. For spaces with fewer than 100 partitions, \( DDName \) is limited six characters. For spaces with 100 or more partitions, \( DDName \) is limited to five characters.

  Table 11 lists the default ddname prefixes.

**Table 11 Prefix defaults for partitioned spaces**

<table>
<thead>
<tr>
<th>Type of copy</th>
<th>Less than 100 partitions</th>
<th>Equal to or greater than 100 partitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>local primary copy</td>
<td>BMCCPY</td>
<td>BMCCY</td>
</tr>
<tr>
<td>local backup copy</td>
<td>BMCCPZ</td>
<td>BMCCZ</td>
</tr>
<tr>
<td>remote primary copy</td>
<td>BMCRCY</td>
<td>BMCRY</td>
</tr>
<tr>
<td>remote backup copy</td>
<td>BMCRCZ</td>
<td>BMCRZ</td>
</tr>
</tbody>
</table>

The copies are made for the space as a whole if OPTIONS OUTCOPY ASCODED is coded, or if the OUTCOPY installation option is defaulted to ASCODED, and you have not coded DSNUM on the RECOVER TABLESPACE, RECOVER INDEXSPACE, or RECOVER INDEX command statement.

For these copies, you specify a ddname of up to eight characters. The default ddnames are the same as those shown in Table 11 in the Less Than 100 Partitions column.

- **DDname Construction for Nonpartitioned Spaces**

  When you are recovering a nonpartitioned space, the construction of the ddname depends on how the recovery is requested. If the request is made for DSNUM ALL (the default), the ddname is the ddname for the copy data set, up to eight characters. The default ddnames are the same as those shown in Table 11 in the Less Than 100 Partitions column.

  If the request is to recover a specific data set of the space, for example DSNUM 2, the ddname becomes a prefix.
The default ddname prefixes depend on the highest DSNUM that you specified in the input commands. If DSNUM is less than 100, the default ddname prefixes are the same as those shown in Table 11 on page 281 in the Less than 100 partitions column.

If the highest DSNUM specified is greater than or equal to 100, a maximum of five characters may be specified for the prefix, and the default values are the same as those shown in Table 11 on page 281 in the Equal to or greater than 100 Partitions column.

Log data sets

Log data sets are used when RECOVER TABLESPACE, RECOVER INDEXSPACE, or RECOVER INDEX command processing involves applying log records. The data sets are dynamically allocated by RECOVER PLUS. No DD statements are allowed for log data sets. To control whether the utility uses archive or active log data sets and which copy is used, refer to OPTIONS RESOURCE SELECTION (see “RESOURCE SELECTION” on page 120).

Log sort work data sets

Log sort work data sets are used when log records are to be applied. For more information, see “Defining sort work data sets” on page 286.

DD statements for REBUILD INDEX jobs

The options specified, or for which defaults are accepted, with the REBUILD INDEX command may require that you specify DD statements in the JCL in addition to those common to all RECOVER PLUS executions (see “DD statements common to all RECOVER PLUS executions” on page 277).

NOTE

Because REBUILD INDEX and RECOVER INDEX are synonyms when you specify INDEXLOG=NO, the information in this section applies to RECOVER INDEX INDEXLOG=NO jobs. It could also apply to RECOVER INDEX INDEXLOG=AUTO jobs if no index image copies are found. (For more information about INDEXLOG, which is an installation option and which can also be set on the OPTIONS statement, see “INDEXLOG=NO” on page 541 and “INDEXLOG” on page 126.) SIMRBLD and SIMRCVR may also require these DD statements.
Use DD statements to specify the following types of data sets:

- sort work data sets for keys
  - parallel sorts
  - non-parallel sorts
- sort message data sets for parallel sorts
- work data sets for extracted keys

If RECOVER UNLOADKEYS command statements exist for the same space in the recovery, see “DD statements for RECOVER UNLOADKEYS jobs” on page 284.

### Data sets needed for parallel index sorts and rebuilds

When you use the MAXKSORT option to specify that multiple key sorts (and index rebuilds) are to run in parallel subtasks, the following data sets are dynamically allocated by RECOVER PLUS, unless you provide DD statements for them:

- **SxxxWKnn**: key sort work files for multiple key sorts running in parallel
  
  \( xxx \) is a number between 1 and the value that you specified for the MAXKSORT installation option or the OPTIONS MAXKSORT parameter. \( nn \) is a number between 1 and the value that you specified for the SORTNUM installation option or the OPTIONS SORTNUM parameter.

- **SYSOU\( nnn \)**: sort message data sets
  
  \( nnn \) is a number between 1 and the value that you specified for the MAXKSORT installation option or the OPTIONS MAXKSORT parameter.

**NOTE**

When you use dynamic allocation for these files, RECOVER PLUS determines the optimal number of files to use.

For more information about MAXKSORT, see “MAXKSORT integer” on page 136, “MAXKSORT” on page 544, “Setting the MAXKSORT option” on page 439, and “Understanding the UNLOADKEYS and BUILDDINDEX strategy” on page 423.
**Sort work data sets for keys for non-parallel sorts**

RECOVER PLUS uses the following sort work data sets for non-parallel sorting of the index keys (when MAXKSORT=1):

- S001WK01 through S001WKnn for key sorts
- SYSOU001 for the sort message data set

For more information, see “Defining sort work data sets” on page 286.

**Work data sets needed for extracted keys (WORKDDN)**

If you code NOWORKDDN, no work data set is used for extracted keys.

If you code WORKDDN, you need to include a DD statement with the same name to specify the work data set. The default ddname is SYSUT1.

The work data set can be shared with other REBUILD INDEX command statements for other spaces. You must use the same work data set for all of the indexes on a single table space or code NOWORKDDN for all of them.

If you code neither WORKDDN nor NOWORKDDN, the use of a key work data set depends on whether you have included a SYSUT1 DD statement in the JCL. If you have included a SYSUT1 DD, it will specify the work data set. If you have not included a SYSUT1 DD, no work data set is used.

**NOTE**

WORKDDN is valid only if MAXKSORT=1. If MAXKSORT is a value greater than 1, processing occurs as if NOWORKDDN is coded.

**DD statements for RECOVER UNLOADKEYS jobs**

You must include DD statements in the JCL when you specify a RECOVER UNLOADKEYS command statement. These DD statements are included in addition to those common to all RECOVER PLUS executions (see page 276). Use DD statements to specify the following data sets:

- work data sets for extracted keys (SKEYDDN)
- (optional) sort work data sets for keys

For examples of DD statements used in a combined RECOVER UNLOADKEYS and REBUILD INDEX job, see “Example 6: Extracting nonpartitioned index keys” on page 365.
Work data set needed for extracted keys (SKEYDDN)

This data set holds keys for subsequent RECOVER BUILINDEX jobs. A sort for this file is done by RECOVER UNLOADKEYS. For the sorted extracted keys, one output data set exists per table space, regardless of how many partitions (or keys) are unloaded in one job.

The ddname is specified by the SKEYDDN option of a RECOVER UNLOADKEYS command. The default is SKEY.

When multiple RECOVER UNLOADKEYS commands are specified for a table space, the same SKEYDDN must be specified in all command statements.

The work data set for a specified table space cannot be shared with other RECOVER UNLOADKEYS commands for other table spaces or other REBUILD INDEX commands for other table spaces.

When you specify REBUILD INDEX and a RECOVER UNLOADKEYS for a table space, NOWORKDDN, which is the default value on the REBUILD INDEX command, sends keys for the partitioned index directly into the sort, bypassing the file held for the subsequent RECOVER BUILINDEX job.

Sort work data sets for keys

These data sets are used for sorting the index keys. Sort work data sets are generally needed if the REBUILD INDEX command is included for the partitioned index. For more information, see “Defining sort work data sets” on page 286.

DD statements for RECOVER BUILINDEX jobs

In addition to those common to all RECOVER PLUS executions (see page 277), DD statements must be included in the JCL when you specify a RECOVER BUILINDEX command statement. Use DD statements for work data sets for extracted keys (SKEYDDN).

For examples of the DD statements used in a RECOVER BUILINDEX job, see “Example 7: Building a nonpartitioned index” on page 366.

Data sets containing extracted keys (SKEYDDN)

RECOVER PLUS expects at least one DD statement that refers to a data set used to collect the sorted keys for nonpartitioned indexes in a prior RECOVER UNLOADKEYS run. If multiple files of sorted keys exist (because the RECOVER UNLOADKEYS command was run in multiple executions to get all of the partitions unloaded and sorted), each file needs a DD statement.

Chapter 4  Building and executing RECOVER PLUS jobs  285
The ddname prefix is specified by the SKEYDDN option of a RECOVER BUILDINDEX command. The default prefix is SKEY. Any ddname within the JCL that begins with the specified prefix is recognized as an unload work data set.

Defining sort work data sets

You can define index sort work data sets in the following ways:

- Let the sort routine make the allocations dynamically by using installation defaults.
- Specify SORTDEVT and SORTNUM and let the sort routine dynamically make the allocations.
- Specify only SORTDEVT, and let BMCSORT determine the number of data sets.
- Code the DD statements in the JCL for the ddnames as follows:
  - L001WK01 through L001WKnn for log sorts (non-parallel)
  - LxxxWKnn for log sorts
    - \(xxx\) is the number of the log sort and is a number between 1 and the value that is specified for MAXLSORT. \(nn\) is the number of the work data set.
  - LOGOU\(nnn\) sort message files
    - \(nnn\) is the number of the log sort and is a number between 1 and the value that is specified for MAXLSORT.
  - S001WK01 through S001WKnn for key sorts (non-parallel)
  - SxxxWK01 through SxxxWKnn
    - \(xxx\) is a number between 1 and the value of the MAXKSORT option and \(nn\) is a number between 1 and the value specified for SORTNUM installation option or the OPTIONS SORTNUM parameter, for key sort work files for multiple index key sorts running in parallel in subtasks.
Determining sort work space requirements

When a recovery involves sorting index keys or sorting the log, you should consider the following requirements:

- virtual memory requirements, which are controlled by the SMCORE installation option (see page 545)

- fixed disk storage (DASD) requirements, which are controlled by the following settings:
  
  — BMCSORT options settings (For more information, see the BMCSORT Reference Manual.)
  
  — parameter settings in the RECOVER PLUS options module (AFR$OPTS), which are described in Appendix A, “RECOVER PLUS installation options”

- number of parallel index key sorts, which is controlled by the MAXKSORT option (page 136 and page 544) and the KSORTSHARE option (page 136 and page 554)

- number of log sorts that can run concurrently

These requirements have an affect on sort performance. For more information, see “Managing sort performance” on page 433 and “Understanding the UNLOADKEYS and BUILDINDEX strategy” on page 423.

Index sort work space requirements

If you need to know the amount of sort work space that will be required for rebuilding indexes, you can request that RECOVER PLUS provide this information. You do this by coding at least one sort work DD statement in your JCL. For example, you could code the following DD statement:

$001WK01 DD DUMMY

You could then run RECOVER PLUS with OPTIONS ANALYZE ONLY to receive a list of sort work data sets that will be used in this job along with the estimated space required for each sort work data set. This output is in the AFRSTMT message data set. An example of the resulting output follows:
Determining sort work space requirements

You can then code DD statements for one or more of the sort work data sets with the amount of disk space given for each data set in the AFRSTMT output. If you do not code DD statements for some of the sort work data sets, BMCSORT dynamically allocates them.

**NOTE**
RECOVER PLUS provides *an approximation only* for the amount of sort work spaces required even if statistics or NUMRECs are accurate. Real space requirements could vary depending on amount of memory available at run time, number and size of sorts running in parallel, the methods of sorts chosen by BMCSORT, and other variables.

**WARNING**
You should use the information generated by the technique described above with caution. If there were significant changes in recoverable objects, used options, environment, and other variables between the ANALYZE ONLY run and real recovery run, RECOVER PLUS could change the recovery plan.

If you do not code any sort work DDs in your JCL (you allow BMCSORT to dynamically allocate all sort work data sets), RECOVER PLUS dynamically selects work files for each sort based on availability, and which set of sort work files are assigned to a particular index is not predictable. In this case, the information in AFRSTMT, as shown in the example output above, is only useful in determining the total amount of required sort work space.

If you code at least one sort work DD in your JCL, RECOVER PLUS statically assigns sort work data sets to indexes during the ANALYZE phase. In this case, you can reliably determine the space requirements for each set of sort work data sets from AFRSTMT, as shown in the example output above.

The estimates that RECOVER PLUS provides are based entirely on the estimated number of keys for each index. RECOVER PLUS obtains this information from the NUMREC parameter if you specify it on the REBUILD statement. Otherwise, RECOVER PLUS uses the following values to determine the estimates:

- For DB2 Version 8 and earlier or for DB2 Version 9 and later with the UT SORT DATA SET ALLOCATION field (DSN6SPRM UTSORTAL subsystem parameter) set to NO (the default value), RECOVER PLUS uses partition or table cardinality if a RUNSTATS operation has been performed for the table space.
For DB2 Version 9 and later with UTSORTAL set to YES, RECOVER PLUS uses real-time statistics by using the TOTALROWS value from table SYSIBM.SYSTABLESPACESTATS for single-table table spaces.

**NOTE**
If UTSORTAL is set to YES and you perform a migration, TOTALROWS may not be accurate. If it is not accurate after the migration, you must reset the value to NULL. However, TOTALROWS is generally accurate and you may not need to reset its value in the following situations:

- if you migrate data daily and the size of the space does not change much
- if you include the real-time statistics tables in your migration

If none of these sources is available, RECOVER PLUS attempts to calculate the number of keys by using the size of the table space data sets. This calculation can be very inaccurate if the table space contains multiple tables or the table contain VARCHAR columns. To get accurate sort work size estimates, you should

- ensure that RUNSTATS is run frequently
- use real-time statistics (by setting UTSORTAL to YES)
- provide an accurate estimate on the NUMREC parameter

The amount of parallelism that RECOVER PLUS can achieve can be affected by coding sort work DDs in the JCL. When you code one or more sort work DDs, RECOVER PLUS statically assigns sorts to indexes, which prevents any overlap of index rebuilds with table space recoveries if keys are being extracted.

For example, assume that you want to recover table space A and table space B and rebuild all indexes for the two table spaces (and MAXLSORT=1). If table space A needs 8 sorts, table space B needs 12 sorts, and MAXKSORT is set to 20, RECOVER PLUS can overlap the building of indexes for table space A with the recovery of table space B. Table space A will use sort work data sets S001WK** through S008WK** and table space B will use sort work data sets S009WK** through S012WK**.

Using this same example, if you code any sort work DDs in your JCL, RECOVER PLUS will use S001WK** through S008WK** for table space A and S001WK** through S012WK** for table space B. When the recovery for table space B begins, the recovery will have to wait for the index builds for table space A to complete because sort work data sets S001WK** through S008WK** are not available.

**NOTE**
BMC recommends that you allow BMCSORT to dynamically allocate all sort work data sets. You should code sort work DD statements in your JCL only if you have a special situation where dynamic allocation will not work.
Log sort space requirements

If a significant number of log records exist, the sort process can require additional system resources. RECOVER PLUS invokes BMCSORT. The amount and type of resources that are required for the sort are dependent on the number of log records to be sorted. The total amount of sort work space that is needed to sort the log optimally is roughly equal to the sum of the lengths of all log records involved in the recovery times two. You can specify L001WKnn DD statements in the JCL for log sort work data sets or use the SORTDEVT and SORTNUM options for RECOVER PLUS.

You can use the MAXLSORT option (page 138 and page 553) to specify that you want to run multiple log sorts.

You can specify LxxxWKnn DD statements in the JCL for log sort work data sets or use the SORTDEVT and SORTNUM options for RECOVER PLUS.

The LOGSCAN command can help you size the log sort work data sets by estimating the number of bytes of log data to be sorted. This command also provides NUMREC and AVGRECSZ estimates that can be used to improve log sort performance.

NOTE

The sorting requirement at recovery time can be reduced significantly by running R+/CHANGE ACCUM in advance. For more information, see the R+/CHANGE ACCUM User Guide.

Determining unloaded keys file space requirements

Use the following information to help estimate the size to allocate for unloaded keys files. Your RUNSTATS must be current, or you must use your own row estimate.

Multiply the number of rows in the table space by the record length to get the bytes needed by RECOVER PLUS, as follows:

\[ \text{Size} = \text{NR} \times \text{RL} \]

In the equation, the variables are defined as follows:

- **Size** is bytes in unloaded keys file.
- **NR** is the number of rows given by the value of CARD from the SYSIBM.SYSTABLEPART table.
Specifying RECOVER PLUS syntax

Use Chapter 3, “RECOVER PLUS syntax” and Chapter 5, “Examples of RECOVER PLUS jobs” as guides to the syntax that is required for completion of your job setup. Place recovery commands in the SYSIN data set.

Running RECOVER PLUS jobs

Running the RECOVER PLUS utility may include the following tasks:

- starting RECOVER PLUS as a batch job
- restarting RECOVER PLUS from the last phase or sync point
- displaying the status of a RECOVER PLUS job
- terminating a RECOVER PLUS job

To run a RECOVER PLUS job, you must have the proper authorizations. For more information, see “Authorization” on page 80.

If a RECOVER PLUS job abnormally terminates, you must clean up the BMCUTIL and BMCSYNC tables as described in “Cleaning up the BMCUTIL and BMCSYNC tables” on page 297. A RECOVER UNLOADKEYS job also requires some cleanup as discussed in “Cleaning up after a RECOVER UNLOADKEYS job” on page 297.

Starting a RECOVER PLUS job

You normally invoke RECOVER PLUS as a batch job by specifying execution of the module AFRMAIN on the EXEC statement of your JCL, with required utility parameters. Some DD statements may also be necessary. For more information, see “RECOVER PLUS data sets and RECOVER PLUS DD statements” on page 276.
Restarting a RECOVER PLUS job

You can restart a RECOVER PLUS job that fails to complete successfully. If checkpoints were taken in the original run, processing phases will not be executed again during the restart.

The value that you specify for the restart parameter (page 271) controls the restarting of the utility. You can restart the utility from the last recorded checkpoint by specifying NEW/RESTART or RESTART for the restart parameter value.

If a table space, index, or index space was marked unrecoverable because of unavailable recovery resources, RECOVER PLUS does not attempt to recover that object in a restarted job. If the recovery resources have been made available since the original job, the objects should be recovered by using the NEW restart parameter.

**NOTE**

When you perform a RESTART, RECOVER PLUS does not attempt to recover unavailable resources, but RECOVER PLUS does issue messages in the planning phase indicating that the resources are not there and gives the appropriate return code.

Table 12 shows the kinds of restart that are possible for each phase of RECOVER PLUS.

**Table 12  Restarts allowed for each phase of RECOVER PLUS**

<table>
<thead>
<tr>
<th>Last incomplete phase</th>
<th>Phase restart allowed?</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTILINIT/ANALYZE</td>
<td>yes</td>
</tr>
<tr>
<td>LOG INPUT</td>
<td>yes</td>
</tr>
<tr>
<td>SNAP</td>
<td>yes</td>
</tr>
<tr>
<td>MERGE&lt;sup&gt;a&lt;/sup&gt;</td>
<td></td>
</tr>
<tr>
<td>RESTORE</td>
<td>yes</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>yes</td>
</tr>
<tr>
<td>BUILD&lt;sup&gt;b&lt;/sup&gt;</td>
<td></td>
</tr>
<tr>
<td>UTILTERM</td>
<td>yes</td>
</tr>
</tbody>
</table>

<sup>a</sup> If you specify restart for a job that failed in a MERGE phase, restart may occur at the beginning of the LOG INPUT phase. However, recoverable objects that were successfully merged will not be remerged during the restart job. True phase restart at such a MERGE phase is possible only for a merge with no log records to be sorted.
To restart a failed job successfully, you must use the same utility ID as in the original run. Also, you must ensure that the commands in the SYSIN data set are compatible with those in the original job. For example, you can remove objects (table spaces and indexes) from the job, but you cannot add objects, nor can you change critical attributes of the job or its objects. The values for ANALYZE must remain the same. None of the index commands can be changed to another type. For example, REBUILD INDEX cannot be changed to RECOVER UNLOADKEYS. Ensure that none of the objects involved in the recovery have been altered between the time the original job terminated and the time of the restart.

The following must apply in the original and restarted jobs:

- For each RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX command in the job, LOGAPPLY ONLY, LOGONLY, FROMRBA/FROMLOGPOINT, TORBA/TOLOGPOINT, TOCOPY, and INDEP OUTSPACE (and the associated data sets) must remain the same.

- For each REBUILD INDEX command, the values of WORKDDN (or NOWORKDDN) and INDEP OUTSPACE (and the associated data sets) must remain the same.

- For RECOVER UNLOADKEYS commands, the values of PART and SKEYDDN must remain the same.

- For the RECOVER BUILDINDEX commands, the value of SKEYDDN must remain the same.

To ensure restartability during the BUILD phase for rebuilding an index, specify DISP=(MOD,CATLG,CATLG) in the DD statement for each key work data set. If you must restart the run, the key work data set will be referred to as DISP=OLD by the program. This DISP value results in correct handling during restart. A subsequent step can delete the work data sets conditionally.

**NOTE**

You cannot restart a RECOVER PLUS job that specifies the ANALYZE ONLY option or the SIMULATE YES option. You must start a new job.
**Restart considerations when creating output copies**

Whether you restart your RECOVER PLUS jobs from the beginning or restart from the point of failure plays a role in how your restart jobs should be set up.

**Restart from the beginning**

If you always plan to restart failed RECOVER PLUS jobs from the beginning, use the NEW restart parameter in the original job. If you allocate the output copy data sets in the JCL, you must also do the following things:

- use DISP=(NEW, CATLG, CATLG) or DISP=(NEW, KEEP, KEEP)
- use unique data set names for each execution
  
  GDGs are helpful for accomplishing this task.
- code BLKSIZE=0 for disk data sets so that unopened data sets can be migrated successfully, if necessary

This method leaves empty, unused data sets if disk copies are made for any object not recovered by the failing run.

The advantage of this method is that only the data set names must be modified for the restart. If GDGs are used, however, no modification of the JCL is necessary. Do not use this method under a restart package that modifies the data set names or dispositions on restart.

**Restart from the point of failure**

If you plan always to restart failed RECOVER PLUS jobs from the point of failure, use the NEW/RESTART restart parameter in the original job. If you allocate copy data sets in the JCL, you must also

- use DISP=(MOD, CATLG, CATLG) or DISP=(NEW, KEEP, KEEP)
- perform the following additional steps at restart:
  
  — For stacked tape copies that use GDGs, modify the data set names to indicate the generation relative to the restart. Modify the 
  
  \((+n)\)
  
  value to 
  
  \((+n-m)\), where 
  
  \(m\) is the relative generation number for the last cataloged generation in the original execution.

  — For cataloged stacked tape copies, remove VOL=REF= from the copy data set DD statements for the copy creation that failed. This removal tells the system to use the catalog for volume information.
Restarting a RECOVER PLUS job

Failure to remove VOL=REF= causes the restarted data set to get a “not cataloged” message and causes a multiple volume data set to be on a different set of volumes than the original, cataloged data set. If the restarted copy data sets expand to more volumes than were cataloged at the time of the original execution, any attempt to stack further data sets by using VOL=REF= results in another abend because the reference uses the catalog information from the beginning of the job step. The system will catalog the expanded data sets again at the end of the job step. However, submitting the job a third time should result in the utility executing with the volumes resolved correctly.

— For uncataloged stacked tape copies, you must include the VOL=SER information of completed copies in the DD statements before restarting. You must also change the NEW disposition to OLD.

This method allows processing to continue with the failed command, minimizing unnecessary processing. This method usually requires manual intervention during restart to modify the JCL.

NOTE

The use of the output copy dynamic allocation feature eliminates the need to modify the JCL during restart. (For more information, see “Allocating output image copy data sets dynamically” on page 452.)

Restart considerations when using TRANSFORM

If the job fails performing an index transformation, the restart will reprocess the entire table space along with the index, even if the table space transformation completed successfully. If the target table space is a partitioned-by-range (PBR) space and the job fails while processing a partition, the restart will not reprocess the partitions that have completed successfully unless the indexes are also being transformed. In this case, the restart will reprocess all the table space partitions. If the target table space is a partition-by-growth (PBG) space, the restart will reprocess the entire table space.

RECOVER PLUS installation options that affect restart—CHECKPT and CHECKINT

Two installation options that affect the restart processing for RECOVER PLUS jobs are CHECKPT and CHECKINT. These options are provided to minimize the time that is lost redoing work when a job fails and must be restarted. The CHECKPT and CHECKINT installation options are described in Appendix A, “RECOVER PLUS installation options.”
You can specify the CHECKPT installation option to take the following checkpoints:

- take no checkpoints
- take checkpoints only at the end of each phase

You can override the CHECKPT installation option at run time by specifying the RECOVER PLUS checkpoint override parameter that is described on “Checkpoint override parameter (checkPt)” on page 273.

**NOTE**

If no checkpoints are requested in the installation option, RECOVER PLUS will not take checkpoints, except those necessary to synchronize RECOVER PLUS execution with the execution of other BMC utilities, as well as MERGE checkpoints that are necessary to guarantee the integrity of output copy, point-in-time recovery, or index rebuild registrations.

When you request checkpoints, you can prevent unnecessary checkpoints by specifying a value for the checkpoint interval installation option, CHECKINT, which sets the number of minutes elapsed between checkpoints. You can override the CHECKINT installation option at run time by specifying CHECKINT on the OPTIONS command, as described on page 117.

To determine values for either or both of these options, you must balance the cost of taking checkpoints against the time that is lost redoing work when a RECOVER PLUS execution must be restarted.

### Restart considerations with INDEXLOG AUTO

When RECOVER PLUS converts a RECOVER INDEX request to a REBUILD INDEX request based on the specification of INDEXLOG AUTO, RECOVER PLUS updates the BMCSYNC table for the index to indicate that it is now a REBUILD INDEX. During restart, RECOVER PLUS uses the BMCSYNC table update to discover that an index recovery was converted.

If a RECOVER INDEX request that uses an image copy fails during execution, the restart logic causes the index to become a candidate for conversion because the original image copy is invalid.

If a RECOVER INDEX DSNUM ALL request for a nonpartitioned index is converted to multiple DSNUM n requests, and if one of the resulting MERGE phases later fails, restart does not allow the request to be converted to a REBUILD INDEX request. If a multi-data-set index is recovered with a DSNUM 0 image copy, the RECOVER INDEX request can convert to a REBUILD INDEX request.
Displaying the status of a RECOVER PLUS job

You can determine the status of RECOVER PLUS jobs in progress or awaiting restart by issuing an SQL SELECT statement on the BMCUTIL table. For a sample statement, see “To display BMC utility status” on page 568.

Terminating a RECOVER PLUS job

You can terminate an active recovery job by deleting the corresponding row from the BMCUTIL table. (For a description of the BMCUTIL table, see Appendix C, “BMC utilities database.”) If the job is in progress, it terminates at the next checkpoint with a return code of 8.

If you require immediate termination, you must cancel the job with the operating system CANCEL command.

If the RECOVER PLUS utility is terminated during any phase except UTILINIT, ANALYZE, or UTILTERM, the table spaces or indexes being recovered may be unusable. When RECOVER PLUS terminates abnormally, spaces are left in STOP (stopped) status. However, spaces that are not recovered may be left in RECP (recovery pending) status, RBDP (rebuild pending) status, PSRCP (page set recovery pending) status, or PSRBD (page set rebuild) status.

Cleaning up the BMCUTIL and BMCSYNC tables

You can clean up after a RECOVER PLUS run terminates abnormally by executing RECOVER PLUS with the same utility ID and specifying TERM as the value of the restart parameter. This cleanup deletes all rows for the utility ID in the BMCUTIL and BMCSYNC tables.

Cleaning up after a RECOVER UNLOADKEYS job

Successful completion of a RECOVER UNLOADKEYS job leaves rows in the BMCSYNC table with blank utility IDs for the table space partitions and the indexes for which keys are being unloaded. If you decide not to run RECOVER BUILDINDEX after a RECOVER UNLOADKEYS, you can remove these rows with the SQL DELETE statement in “Cleaning up RECOVER UNLOADKEYS entries” on page 578.
Using MSGLEVEL to control RECOVER PLUS output

You can control the amount of output that RECOVER PLUS produces based on the value that you specify for the MSGLEVEL utility parameter (see page 273). Valid values of MSGLEVEL are 0, 1, and 2, with a default value of 1. The example in Figure 35 was run by using each value of MSGLEVEL. Different output files are received, and different information is in the output files, for each run. The files that are produced are based on the different values of MSGLEVEL as summarized in Table 10 on page 277.

**Figure 35  RECOVER PLUS JCL for recovery of table space and index**

```plaintext
//AFREXMLx JOB (PAFR),'EXAMPLE MSGLEVEL',
//     CLASS=Q,NOTIFY=&SYSUID,
//     MSGCLASS=X
//--
//* THIS JOB RECOVERS A TABLE SPACE AND REBUILDS THE INDEX TO CURRENT.
//* EXAMPLE OF MSGLEVEL(x) OUTPUT.
//*-------------------------------------------------------------------
//*RECOVER EXEC PGM=AFRMAIN,REGION=0M,
//     PARM='DGE,AFRMSGLx,NEW/RESTART,MSGLEVEL(x)' x set to 0, 1, or 2
//STEPLIB DD DISP=SHR,DSN=product.libraries
//     DD DISP=SHR,DSN=DB2.DSNEXIT
//     DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN DD *
OPTION EARLYRECALL INDEXLOG NO
RECOVER TABLESPACE BMCDBSMP.BMCTS014
REBUILD INDEX (RDASTC.BMCIX014)
/*
//
```

**Sample output with MSGLEVEL(0)**

When you specify MSGLEVEL(0), RECOVER PLUS produces the following output files:

- AFRSUMRY (Figure 36 on page 299)
- AFR_STMT (Figure 37 on page 299)
- AFRPRINT (Figure 38 on page 301)
Figure 36  AFRSUMRY with MSGLEVEL(0)

BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40001I UTILITY EXECUTION STARTING 01/09/2011 15:57:54

BMC40876I MAINT:  NO RECOVER PLUS PTFS APPLIED
BMC40876I MAINT:  BPJ0062  BPJ0065  BPJ0067  BPJ0071  BPJ0079  BPJ0081  BPJ0087
BMC40876I MAINT:  BPJ0088  BPJ0089  BPJ0091  BPJ0098  BPJ0105  BPJ0107  BPJ0109
BMC40876I MAINT:  BPJ0110  BPJ0111  BPJ0113  BPJ0116  BPJ0119  BPJ0126  BPJ0127
BMC40876I MAINT:  BPJ0137  BPJ0142  BPJ0144  BPJ0150  BPJ0152  BPJ0155

BMC40002I UTILITY ID = AFRMSGL0. DB2 SUBSYSTEM ID = DGA1.

BMC40094I ATTEMPTING TO CONNECT TO DB2 SUBSYSTEM DGA1, USING PLAN AFRR101T
BMC400067I DB2 DGA1 IS MEMBER DGA1 IN DATA SHARING GROUP DSNDGA

BMC40475I INPUT STATEMENTS PRINTED IN AFRSTMT

BMC40336I ANALYZE FINISHED.
BMC40287I ELAPSED TIME = 00:00:03. UTILITY ELAPSED TIME = 00:00:05
BMC40288I CPU TIME = 00:00:00. UTILITY CPU TIME = 00:00:00

BMC40336I PLANNING FINISHED.
BMC40287I ELAPSED TIME = 00:00:00. UTILITY ELAPSED TIME = 00:00:05
BMC40288I CPU TIME = 00:00:00. UTILITY CPU TIME = 00:00:00

BMC40336I PLAN EXECUTION FINISHED.
BMC40287I ELAPSED TIME = 00:00:04. UTILITY ELAPSED TIME = 00:00:10
BMC40288I CPU TIME = 00:00:00. UTILITY CPU TIME = 00:00:00

BMC40476I EXECUTION SUMMARY
BMC40477I TABLE SPACES (PARTITIONS) RECOVERED   :  1 (0)
BMC40479I LOG RECORDS APPLIED                   :  9995
BMC40480I INDEXES REBUILT                       :  1

BMC40005I UTILITY EXECUTION COMPLETE 01/09/2011 15:58:04, RETURN CODE = 0

Figure 37  AFRSTMT with MSGLEVEL(0) (part 1 of 3)

BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40937I z/OS RELEASE = 1.8.00, PID = HBB7730, SMS RELEASE = 1.8.0
BMC40157I AVAILABLE REGION BELOW 16M = 8872K, AVAILABLE REGION ABOVE 16M = 1590048K, NUMBER OF CPUS = 5
BMC40931I PARM LIST:   DB2 SSID = DGA1
BMC40932I UTILID = AFRMSGL0
BMC40009I UTILITY NEW/RESTART REQUESTED - SYNCPOINT RESTART WILL OCCUR IF UTILID IS IN BMCUTIL
BMC40933I RESTART = NEW/RESTART
BMC40934I MSGLEVEL = 0
BMC40935I PARM NOT CODED - CHECKPT
BMC40936I PARM NOT CODED - RDB2STAT
BMC40937I PARM NOT CODED - AFROPTS
BMC40938I PARM NOT CODED - ACAOPTS
BMC40924I OPTIONS:     PLANRECV = AFRR101T       (MAIN PLAN)
BMC40925I TBUFFS = 1000 (BUFFER MANAGER PAGES)
BMC40944I OPNDB2ID = YES (ACQUIRE RACF AUTHORITY OF DB2)
BMC40926I CHECKPT = PHASE (CHECKPOINTING)
BMC40945I CHECKINT = 0 (CHECKPOINT INTERVAL MINUTES)
BMC40927I RDB2STAT = YES (RESET DB2 OBJECT STATUS)
BMC40946I AMSCAT = NO (USE 'CATALOG' IN IDCAMS INPUT)
BMC40928I SMORE = OK, OK (SORT CORE VALUES)
BMC40931I RESINV = OK (MEMORY BELOW 16MB EXCLUDED FROM USE BY SORT)
<table>
<thead>
<tr>
<th>Message Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40938I</td>
<td>WKUNIT = SYSALLDA (WORK UNIT NAME)</td>
</tr>
<tr>
<td>BMC40956I</td>
<td>RCLTSK = 10 (MAXIMUM RECALL TASKS)</td>
</tr>
<tr>
<td>BMC40941I</td>
<td>IXRECP = NO (SET INDEX RECOVER PENDING)</td>
</tr>
<tr>
<td>BMC40891I</td>
<td>MAXDRIVES = 0 (DEFAULT MAX TAPE DRIVES)</td>
</tr>
<tr>
<td>BMC40892I</td>
<td>MAXLOGS = 5 (DEFAULT MAX CONCURRENT LOG FILES)</td>
</tr>
<tr>
<td>BMC40942I</td>
<td>MAXSORT = 10 (DEFAULT) (MAX CONCURRENT INDEX Sorts)</td>
</tr>
<tr>
<td>BMC40962I</td>
<td>SORTDEV = SORT DEFAULT (DEVICE TYPE FOR LOG AND INDEX Sorts)</td>
</tr>
<tr>
<td>BMC40963I</td>
<td>SORTNUM = SORT DEFAULT (WORK DATASETS FOR LOG AND INDEX Sorts)</td>
</tr>
<tr>
<td>BMC40964I</td>
<td>OUTCOPY = ASCODED (OUTCOPY CREATION DIRECTIVE)</td>
</tr>
<tr>
<td>BMC40547I</td>
<td>INDEXLOG = NO (DEFAULT USE OF INDEXLOG STRATEGY)</td>
</tr>
<tr>
<td>BMC40891I</td>
<td>MAXDRIVES = 0 (DEFAULT MAX TAPE DRIVES)</td>
</tr>
<tr>
<td>BMC40892I</td>
<td>MAXLOGS = 5 (DEFAULT MAX CONCURRENT LOG FILES)</td>
</tr>
<tr>
<td>BMC40941I</td>
<td>IXRECP = NO (SET INDEX RECOVER PENDING)</td>
</tr>
<tr>
<td>BMC40942I</td>
<td>MAXSORT = 10 (DEFAULT) (MAX CONCURRENT INDEX Sorts)</td>
</tr>
<tr>
<td>BMC40962I</td>
<td>SORTDEV = SORT DEFAULT (DEVICE TYPE FOR LOG AND INDEX Sorts)</td>
</tr>
<tr>
<td>BMC40963I</td>
<td>SORTNUM = SORT DEFAULT (WORK DATASETS FOR LOG AND INDEX Sorts)</td>
</tr>
<tr>
<td>BMC40964I</td>
<td>OUTCOPY = ASCODED (OUTCOPY CREATION DIRECTIVE)</td>
</tr>
<tr>
<td>BMC40547I</td>
<td>INDEXLOG = NO (DEFAULT USE OF INDEXLOG STRATEGY)</td>
</tr>
<tr>
<td>BMC40537I</td>
<td>ERRCONT = 10 (DEFAULT MAX SEVERE ERRORS)</td>
</tr>
<tr>
<td>BMC40521I</td>
<td>AUTOSIZE = YES (AUTO SIZE ACCUM OUTPUT FILES)</td>
</tr>
<tr>
<td>BMC40241I</td>
<td>PUBLICPLAN = YES (GRANT PLAN TO PUBLIC)</td>
</tr>
<tr>
<td>BMC40081I</td>
<td>BINDQUALIFIER = BMCAFTR (QUALIFIER USED FOR DYNAMIC BIND)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE01 = 3590-1</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE02 = 3490</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE03 = 3480X</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE04 = CART</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE05 = CARTVTS</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE06 = CRTNS</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE07 = CRTNS</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE08 = IBMATL</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE09 = IBMVTS</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE10 = TAPEABL</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE11 = 3480</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE12 = 3400-9</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE13 = SYS3480R</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE14 = SYS3480XR</td>
</tr>
<tr>
<td>BMC40929I</td>
<td>SORT VALUES: SORTCORE = 0 K (SORT MEMORY VALUE)</td>
</tr>
<tr>
<td>BMC40947I</td>
<td>VSCORET = 0 K (TOTAL MEMORY OVERRIDE)</td>
</tr>
<tr>
<td>BMC96111I</td>
<td>DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE</td>
</tr>
<tr>
<td>BMC96131I</td>
<td>COPIES = (LP,LB) (DEFAULT SEQUENCE FOR IMAGE COPIES)</td>
</tr>
<tr>
<td>BMC96141I</td>
<td>ACCUMS = (LP,LB) (DEFAULT SEQUENCE FOR CA FILES)</td>
</tr>
<tr>
<td>BMC96121I</td>
<td>DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE</td>
</tr>
<tr>
<td>BMC96131I</td>
<td>COPIES = (RP,RB) (DEFAULT SEQUENCE FOR IMAGE COPIES)</td>
</tr>
<tr>
<td>BMC96141I</td>
<td>ACCUMS = (RP,RB) (DEFAULT SEQUENCE FOR CA FILES)</td>
</tr>
<tr>
<td>BMC40876I</td>
<td>MAINT: NO RECOVER PLUS PTFS APPLIED</td>
</tr>
<tr>
<td>BMC96010I</td>
<td>SOLUTION COMMON CODE V10.1.01</td>
</tr>
<tr>
<td>BMC40876I</td>
<td>MAINT: NO SOLUTION COMMON CODE PTFS APPLIED</td>
</tr>
<tr>
<td>BMC40037I</td>
<td>UTILID AFMSGL0 WAS NOT FOUND IN THE BMCUTIL TABLE - UTILITY WILL RUN AS IF 'NEW' WERE SPECIFIED</td>
</tr>
<tr>
<td>BMC40611I</td>
<td>INPUT STATEMENTS:</td>
</tr>
<tr>
<td>BMC40101I</td>
<td>OPTION EARLYRECALL INDEXLOG NO</td>
</tr>
<tr>
<td>BMC40101I</td>
<td>RECOVER TABLESPACE BMDCBSMP.BMCTS014</td>
</tr>
<tr>
<td>BMC40101I</td>
<td>REBUILD INDEX (ROAWNW.BMICX014)</td>
</tr>
<tr>
<td>BMC40101I</td>
<td>REBUILD INDEX (ROAWNW.BMICX014)</td>
</tr>
<tr>
<td>BMC40101I</td>
<td>INDEXLOG STRATEGY DISABLED</td>
</tr>
<tr>
<td>BMC4035I</td>
<td>ANALYZE STARTING 01/09/2011 15:57:56</td>
</tr>
<tr>
<td>BMC4035I</td>
<td>CATALOG LOOKUP STARTING 01/09/2011 15:57:56</td>
</tr>
<tr>
<td>BMC40068I</td>
<td>THE KEYS FOR INDEX ROAWNW.BMICX014 WILL GO DIRECTLY TO THE SORT BECAUSE THERE IS NO SYSUT1 DD STATEMENT IN THE JCL</td>
</tr>
<tr>
<td>BMC4063I</td>
<td>USING STANDARD DB2 SECURITY</td>
</tr>
<tr>
<td>BMC40705I</td>
<td>VOLUMES IN STOGROUP SYSDEFLT WILL BE USED IN THE FOLLOWING ORDER:</td>
</tr>
<tr>
<td>BMC40707I</td>
<td>(DEVXXX)</td>
</tr>
<tr>
<td>BMC4036I</td>
<td>CATALOG LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC4035I</td>
<td>SYNCHRONIZATION STARTING 01/09/2011 15:57:57</td>
</tr>
<tr>
<td>BMC4036I</td>
<td>SYNCHRONIZATION FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:02, UTILITY ELAPSED TIME = 00:00:05</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
</tbody>
</table>
Sample output with MSGLEVEL(1)

When you specify MSGLEVEL(1), RECOVER PLUS produces the following output files:

- AFRSUMRY, which is identical to the output of MSGLEVEL(0) (Figure 36 on page 299)
- AFRSTMRT, which is similar to the output with MSGLEVEL(0) (Figure 37 on page 299)
- AFRPRINT (Figure 39)
- AFROSUM (Figure 40 on page 303)

Figure 37  AFRSTMT with MSGLEVEL(0) (part 3 of 3)

When you specify MSGLEVEL(1), RECOVER PLUS produces the following output files:

- AFRSUMRY, which is identical to the output of MSGLEVEL(0) (Figure 36 on page 299)
- AFRSTMRT, which is similar to the output with MSGLEVEL(0) (Figure 37 on page 299)
- AFRPRINT (Figure 39)
- AFROSUM (Figure 40 on page 303)
Figure 39  **AFRPRINT with MSGLEVEL(1) (part 2 of 3)**

<table>
<thead>
<tr>
<th>Message</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40954I</td>
<td>AVAILABLE MEMORY BELOW THE LINE = 8620K, TOTAL AVAILABLE MEMORY (AT LEAST) = 1598192K</td>
</tr>
<tr>
<td>BMC40441I</td>
<td>BMCSORT PRODUCT LOCATED AND WILL BE USED</td>
</tr>
<tr>
<td>BMC40020I</td>
<td>SORT FILE SIZE = 8470528, SORT MEMORY = 450880</td>
</tr>
<tr>
<td>BMC40725I</td>
<td>PROCESSING LOG RANGE:</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = DSNDGA.DGA1.LOGCOPY1.DS13.DATA</td>
</tr>
<tr>
<td>BMC40555I</td>
<td>FROMRBA = X’02167B4DE000’  TORBA = X’02167B8E77B1’</td>
</tr>
<tr>
<td>BMC40724I</td>
<td>DATA SHARING MEMBERID IS 1</td>
</tr>
<tr>
<td>BMC40193I</td>
<td>3 LOG DATASET BUFFERS OF 348160 BYTES EACH WERE ALLOCATED</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = DSNDGA.DGA1.LOGCOPY1.DS13.DATA</td>
</tr>
<tr>
<td>BMC40173I</td>
<td>TIME SPENT IN DATASET ALLOCATION = 00:00:00</td>
</tr>
<tr>
<td>BMC40989I</td>
<td>NUMBER OF PAGES OUTPUT IS = 2002</td>
</tr>
<tr>
<td>BMC40981I</td>
<td>FULL IC RDAWHW.BMCDBSMP.BMCTS014.IC.G0002V00 READ WAITS = 7, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40900I</td>
<td>MERGE TABLESPACE STEP COMPLETE</td>
</tr>
<tr>
<td>BMC40012I</td>
<td>LOG INPUT PHASE COMPLETE. ELAPSED TIME = 00:00:01, TIME SINCE UTILITY START = 00:00:06</td>
</tr>
<tr>
<td>BMC40868I</td>
<td>LOG INPUT PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40016I</td>
<td>MERGE PHASE STARTING 01/09/2011 16:02:35</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = BMCDBSMP.BMCTS014</td>
</tr>
<tr>
<td>BMC40793I</td>
<td>EXTRACTING KEYS FOR:</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = RDAWHW.BMCIX014</td>
</tr>
<tr>
<td>BMC40417I</td>
<td>1 VSAM DATA SETS NOW ALLOCATED</td>
</tr>
<tr>
<td>BMC40484I</td>
<td>MESSAGES FROM PHASE SUBTASK:</td>
</tr>
<tr>
<td>BMC40481I</td>
<td>END OF PHASE SUBTASK MESSAGES</td>
</tr>
<tr>
<td>BMC40911I</td>
<td>01/09/2011 16:02:38 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE</td>
</tr>
<tr>
<td>BMC40016I</td>
<td>BUILD PHASE STARTING 01/09/2011 16:02:38</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = RDAWHW.BMCIX014</td>
</tr>
<tr>
<td>BMC40481I</td>
<td>END OF PHASE SUBTASK MESSAGES</td>
</tr>
<tr>
<td>BMC40904I</td>
<td>THIS PHASE WILL RUN IN EXECUTION SUBTASK 1</td>
</tr>
<tr>
<td>BMC40102I</td>
<td>BUILD INDEX STEP STARTING</td>
</tr>
<tr>
<td>BMC40366I</td>
<td>BUFFER CREATED FOR PAGE SIZE 4K, REFRESH=1</td>
</tr>
</tbody>
</table>

---

**Figure 39**

This figure continues the AFRPRINT output with MSGLEVEL(1), showing details of memory usage, log processing range, and statistics from log read and merge phases. The output includes information on available memory, sort file size, and log dataset buffers allocated. Statistics from the log read step are also provided, including file reads, I/O waits, and log records processed. The merge phase statistics include log input and merge steps, along with details on index extraction and merge progress. The build phase statistics highlight index assignment and other phase-specific operations.
Figure 39  AFRPRINT with MSGLEVEL(1) (part 3 of 3)

| BMC40372I | TOTAL WAITS IN BUFFER MANAGER = 1 |
| BMC40373I | RANDOM WRITE WAITS = 0, RANDOM READ WAITS = 0, RANDOM SYNC WAITS = 0 |
| BMC40374I | CYLINDER WAITS = 1, SEQUENTIAL SYNC WAITS = 0 |
| BMC40371I | CYLINDERS WRITTEN = 1 |
| BMC40375I | CYLINDERS READS = 0, HITS CAUSED BY PREFETCH = 0, HITS ON PREFETCH = 0 |
| BMC40376I | CYLINDER WAIT TIME = 00:00:00, WAIT TIME FOR BUFFER MANAGER = 00:00:00 |
| BMC40379I | CYLINDER WAIT TIME = 00:00:00 |
| BMC40770I | BUILD STATISTICS: # INPUT RECORDS = 1999 # KEY VALUES = 1999 |
| BMC40771I | # WAITS FOR SORT = 0 |
| BMC40772I | # GETPAGE CALLS = 25 # PAGES WRITTEN = 11 # LEAF PAGES = 7 |
| BMC40773I | CYLINDER WAIT TIME = 00:00:00 |
| BMC40774I | CYLINDER WAIT TIME = 00:00:00 |
| BMC40775I | CYLINDER WAIT TIME = 00:00:00 |

Figure 40  AFROSUM with MSGLEVEL(1)

| BMC40018I | RECOVER PLUS FOR DB2 V10.1.00 |
| BMC40473I | COPYRIGHT BMC SOFTWARE INC. 1991-2011 |
| BMC40617I | RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884 |
| BMC40641I | CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM |

BMC40709I OBJECT SUMMARY:

BMC40301I TABLESPACE = BMCDBSMP.BMCTS014
BMC40711I DBID = 'X'001D'
BMC40712I ORIGINAL STATUS = RW
BMC40716I STOGROUP = SYSDEFLT PRIQTY = 800 SECQTY = 200
BMC40710I PAGE SIZE = 4K SEGSIZE = 4
BMC40723I ENCODING SCHEME = EBCDIC

BMC40585I FROM LOGPOINT = 'X'CODE987E5735' TO LOGPOINT = 'X'CODE409E73E'
BMC40727I DISCRETE LOG RANGE(S):
BMC40559I FROMRBA = 'X'02167B4DE4AF' TORMBA = 'X'02167B857BCC'
BMC40555I FROMRBA = 'X'02167B85A30B' TORMBA = 'X'02167B8DD7B1'

BMC40840I INPUT COPY / SYSCOPY DATA:
BMC40620I 2011-01-09 13.59.16.7540 RBA/LRSN 'X'CODE987E5735' FULL IMAGE COPY
BMC40640I 2011-01-09 13.59.18.2802 RBA/LRSN 'X'CODE98807C81' QUIESCE UTILITY POINT - WRITE(YES)

BMC40709I OBJECT SUMMARY:

BMC40346I INDEXSPACE = BMCDBSMP.BMCIX014
BMC40711I DBID = 'X'001D' PSID = 'X'0020'
BMC40712I ORIGINAL STATUS = RW
BMC40303I INDEX = RDAWHW.BMCIX014
BMC40835I PAGE SIZE = 4K
BMC40830I ON TABLE RDAWHW.BMCTB014 (OBID 'X'001E') IN TABLESPACE BMCDBSMP.BMCTS014
BMC40836I TABLE CARDINALITY = -1
BMC40832I NON-CLUSTERING INDEX
BMC40844I TYPE 2
BMC40865I INDEX REBUILD - GROUP 1
When you specify MSGLEVEL(2), RECOVER PLUS produces the following output files:

- AFRSUMRY, which is identical to the output of MSGLEVEL(0) (Figure 36 on page 299)
- AFRSTMT (Figure 41)
- AFRPRINT (Figure 42 on page 307)
- AFROSUM, which is identical to the output of MSGLEVEL(1) (Figure 40 on page 303)
- AFRPLAN (Figure 43 on page 309)
Figure 41  AFRSTMT with MSGLEVEL(2) (part 2 of 4)

<table>
<thead>
<tr>
<th>BMC409671</th>
<th>TAPE DEVICE06 = CRT8NS</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC409671</td>
<td>TAPE DEVICE07 = CRT9NS</td>
</tr>
<tr>
<td>BMC409671</td>
<td>TAPE DEVICE08 = IBMATL</td>
</tr>
<tr>
<td>BMC409671</td>
<td>TAPE DEVICE09 = IBMVTS</td>
</tr>
<tr>
<td>BMC409671</td>
<td>TAPE DEVICE10 = TAPEABL</td>
</tr>
<tr>
<td>BMC409671</td>
<td>TAPE DEVICE11 = 3400-9</td>
</tr>
<tr>
<td>BMC409671</td>
<td>TAPE DEVICE12 = SYS3480R</td>
</tr>
<tr>
<td>BMC409671</td>
<td>TAPE DEVICE13 = SYS348XR</td>
</tr>
<tr>
<td>BMC409671</td>
<td>TAPE DEVICE14 = SYS348KR</td>
</tr>
<tr>
<td>BMC409291</td>
<td>SORT VALUES: SORTCORE = 0 K (SORT MEMORY VALUE)</td>
</tr>
<tr>
<td>BMC409471</td>
<td>VSORET = 0 K (TOTAL MEMORY OVERRIDE)</td>
</tr>
<tr>
<td>BMC961111</td>
<td>DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE</td>
</tr>
<tr>
<td>BMC961131</td>
<td>COPIES = (LP,LB) (DEFAULT SEQUENCE FOR IMAGE COPIES)</td>
</tr>
<tr>
<td>BMC961141</td>
<td>ACCUMS = (LP,LB) (DEFAULT SEQUENCE FOR CA FILES)</td>
</tr>
<tr>
<td>BMC961121</td>
<td>DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE</td>
</tr>
<tr>
<td>BMC961131</td>
<td>COPIES = (RP,RB) (DEFAULT SEQUENCE FOR IMAGE COPIES)</td>
</tr>
<tr>
<td>BMC961141</td>
<td>ACCUMS = (RP,RB) (DEFAULT SEQUENCE FOR CA FILES)</td>
</tr>
<tr>
<td>BMC408761</td>
<td>MAINT: NO RECOVER PLUS PTFs APPLIED</td>
</tr>
<tr>
<td>BMC4060101</td>
<td>SOLUTION COMMON CODE V10.1.00</td>
</tr>
<tr>
<td>BMC408761</td>
<td>MAINT: NO SOLUTION COMMON CODE PTFs APPLIED</td>
</tr>
<tr>
<td>BMC403351</td>
<td>PACKAGE CHECKING AND BINDING STARTING 01/09/2011 16:13:40</td>
</tr>
<tr>
<td>BMC403361</td>
<td>PACKAGE CHECKING AND BINDING FINISHED.</td>
</tr>
<tr>
<td>BMC403281</td>
<td>ELAPSED TIME = 00:00:01, UTILITY ELAPSED TIME = 00:00:01</td>
</tr>
<tr>
<td>BMC402881</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409541</td>
<td>AVAILABLE MEMORY BELOW THE LINE = 8848K, TOTAL AVAILABLE MEMORY (AT LEAST) = 1598436K</td>
</tr>
<tr>
<td>BMC400201</td>
<td>BYT:0, 0, 0; BLK:0, 0, 0; TOT:1858304, 5950596, 5950596; iter: 1; RC:4</td>
</tr>
<tr>
<td>BMC400371</td>
<td>UTLID AFRMSGL2 WAS NOT FOUND IN THE BMCUTIL TABLE - UTILITY WILL RUN AS IF 'NEW' WERE SPECIFIED</td>
</tr>
<tr>
<td>BMC406111</td>
<td>INPUT STATEMENTS:</td>
</tr>
<tr>
<td>BMC401011</td>
<td>OPTION EARLYRECALL INDEXLOG NO</td>
</tr>
<tr>
<td>BMC401011</td>
<td>RECOVER TABLESPACE BMDBSP.BMCTS014</td>
</tr>
<tr>
<td>BMC401011</td>
<td>REBUILD INDEX (RDAWHW.BMCI014)</td>
</tr>
<tr>
<td>BMC401011</td>
<td>INDEXLOG STRATEGY DISABLED</td>
</tr>
<tr>
<td>BMC403351</td>
<td>ANALYZE STARTING 01/09/2011 16:13:42</td>
</tr>
<tr>
<td>BMC403351</td>
<td>CATALOG LOOKUP STARTING 01/09/2011 16:13:42</td>
</tr>
<tr>
<td>BMC403361</td>
<td>PROC RO STMT FINISHED.</td>
</tr>
<tr>
<td>BMC402871</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC402881</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC403361</td>
<td>PROC IX STMT FINISHED.</td>
</tr>
<tr>
<td>BMC402871</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC402881</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC403361</td>
<td>LOOKUP SYSIX FINISHED.</td>
</tr>
<tr>
<td>BMC402871</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC402881</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC403361</td>
<td>SCAN SYSKY FINISHED.</td>
</tr>
<tr>
<td>BMC402871</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC402881</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC403361</td>
<td>COL LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC402871</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC402881</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC403361</td>
<td>SCAN SYSFL FINISHED.</td>
</tr>
<tr>
<td>BMC402871</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC402881</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC400681</td>
<td>THE KEYS FOR INDEX RDAWHW.BMCI014 WILL GO DIRECTLY TO THE SORT BECAUSE THERE IS NO SYSUT1 DD STATEMENT IN THE JCL</td>
</tr>
<tr>
<td>BMC403361</td>
<td>TS LOOKUP FINISHED.</td>
</tr>
</tbody>
</table>
### Figure 41  AFRSTMT with MSGLEVEL(2) (part 3 of 4)

<table>
<thead>
<tr>
<th>Message Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td></td>
<td>USING STANDARD DB2 SECURITY</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>DB LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>TB LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>IP LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>SG LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40705I</td>
<td>VOLUMES IN STOGROUP SYSEFLT WILL BE USED IN THE FOLLOWING ORDER:</td>
</tr>
<tr>
<td></td>
<td>(DEVXXX)</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>SCAN SYSVOL FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>CHK DS FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>HDR PG RD FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>SYSCOPY LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>CATALOG LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40335I</td>
<td>SYNCHRONIZATION STARTING 01/09/2011 16:13:43</td>
</tr>
<tr>
<td>BMC40025I</td>
<td>OUTPUT FROM COMMAND '-STO DB(BMCDBSMP) SPACE(BMCTS014) AT (COMMIT)' FOLLOWS:</td>
</tr>
<tr>
<td>BMC40020I</td>
<td>DSN9022I  *DGA1 DSNTDDIS 'STOP DATABASE' NORMAL COMPLETION</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>SYNCHRONIZATION FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>DDSCAN FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:02</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40335I</td>
<td>PLAN EXECUTION STARTING 01/09/2011 16:13:46</td>
</tr>
<tr>
<td>BMC40052I</td>
<td>OUTPUT FROM COMMAND '-STO DB(BMCDBSMP) SPACE(BMCTS014) AT (COMMIT)' FOLLOWS:</td>
</tr>
<tr>
<td>BMC40020I</td>
<td>DSN9022I  *DGA1 DSNTDDIS 'STOP DATABASE' NORMAL COMPLETION</td>
</tr>
<tr>
<td>BMC40025I</td>
<td>OUTPUT FROM COMMAND '-STO DB(BMCDBSMP) SPACE(BMCTS014) AT (COMMIT)' FOLLOWS:</td>
</tr>
<tr>
<td>BMC40066I</td>
<td>TIME TO READ SYSLGRNX WAS 00:00:00</td>
</tr>
<tr>
<td>BMC40512I</td>
<td>LOG FILE RESOURCES:</td>
</tr>
<tr>
<td></td>
<td>DSN = DSNDDA.DGA1.LOGCOPY1.DS13.DATA</td>
</tr>
<tr>
<td>BMC40555I</td>
<td>FROMRBA = X’02167B4DE000’ TORBA = X’02167BEE7B1’</td>
</tr>
<tr>
<td>BMC40335I</td>
<td>PLAN EXECUTION STARTING 01/09/2011 16:13:46</td>
</tr>
</tbody>
</table>
Chapter 4 Building and executing RECOVER PLUS jobs

Sample output with MSGLEVEL(2)

Figure 41 AFRSTMT with MSGLEVEL(2) (part 4 of 4)

Figure 42 AFRPRINT with MSGLEVEL(2) (part 1 of 3)
Figure 42  AFRPRINT with MSGLEVEL(2) (part 2 of 3)
Figure 42 AFRPRINT with MSGLEVEL(2) (part 3 of 3)

| BMC40374I | CYLINDER WAITS = 1, SEQUENTIAL SYNC WAITS = 0 |
| BMC40371I | CYLINDERS WRITTEN = 1 |
| BMC40375I | RANDOM READS = 0, HITS = 0 |
| BMC40376I | PREFETCH COUNT = 0, HITS CAUSED BY PREFETCH = 0, HITS ON PREFETCH = 0 |
| BMC40377I | RANDOM Writes = 0 in 0 groups |
| BMC40378I | TOTAL WAIT TIME = 00:00:00, RANDOM WRITE TIME = 00:00:00, RANDOM READ TIME = 00:00:00 |
| BMC40379I | CYLINDER WAIT TIME = 00:00:00 |
| BMC40770I | BUILD STATISTICS: # INPUT RECORDS = 1999 # KEY VALUES = 1999 |
| BMC40771I | # WAITS FOR SORT = 0 |
| BMC40772I | # GETPAGE CALLS = 25 # PAGES WRITTEN = 11 # LEAF PAGES = 7 |
| BMC40919I | TOTAL TIME FOR WAITS = 00:00:00, WAIT TIME FOR BUFFER MANAGER = 00:00:00 |
| BMC40012I | BUILD PHASE COMPLETE. ELAPSED TIME = 00:00:00, TIME SINCE UTILITY START = 00:00:08 |

Figure 43 AFRPLAN with MSGLEVEL(2) (part 1 of 2)

| BMC40018I | RECOVER PLUS FOR DB2 V10.1.00 |
| BMC40473I | COPYRIGHT BMC SOFTWARE INC. 1991-2011 |
| BMC96173I | RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884 |
| BMC40700I | EXECUTION PLAN SUMMARY: |
| BMC40701I | PHASE: LOG INPUT |
| BMC40721I | PHASE INITIALIZATION: |
| BMC40738I | INITIALIZE LOG RECORD SORT |
| BMC40814I | ESTIMATED RECORD SIZE = 1000 |
| BMC40813I | ESTIMATED NUMBER OF RECORDS = 1034 |
| BMC40738I | ALLOCATE 2 POINTER BUFFERS (51736 BYTES EACH) |
| BMC40702I | STEP: READ LOG PAGES |
| BMC40736I | STEP INITIALIZATION: |
| BMC40738I | ALLOCATE INPUT LOG FILE |
| BMC40302I | DSN = DSNDGA.DGA1.LOGCOPY1.DS13.DATA |
| BMC40738I | FILE OPTIMIZATION |
| BMC40738I | OPEN LOG DATASET |
| BMC40302I | DSN = DSNDGA.DGA1.LOGCOPY1.DS13.DATA |
| BMC40738I | ALLOCATE LOG DATASET BUFFER(S) |
| BMC40738I | PREPARE LOG FOR PROCESSING |
| BMC40302I | DSN = DSNDGA.DGA1.LOGCOPY1.DS13.DATA |
| BMC40704I | 1034 PAGES FROM PAGE NUMBER 10125 |
| BMC40738I | CLOSE LOG DATASET |
| BMC40302I | DSN = DSNDGA.DGA1.LOGCOPY1.DS13.DATA |
| BMC40738I | DEALLOCATE INPUT LOG FILE |
| BMC40302I | DSN = DSNDGA.DGA1.LOGCOPY1.DS13.DATA |
| BMC40738I | FREE 3 LOG DATASET BUFFER(S) |
| BMC40737I | STEP TERMINATION: |
| BMC40735I | PHASE TERMINATION: |
| BMC40738I | END LOG RECORD SORT INPUT |
| BMC40738I | FREE POINTER BUFFER |

| BMC40011I | Phrase: MERGE |
| BMC40721I | PHASE INITIALIZATION: |
| BMC40301I | TABLESPACE = BMCDBSMP.BMCTS014 |
| BMC40793I | EXTRACTING KEYS FOR: |
| BMC40303I | INDEX = RDAWHW.BMCIX014 |
| BMC40738I | ALLOCATE (2) KEY BUFFER(S) (270336 BYTES EACH) FOR INDEX GROUP 1 |
| BMC40738I | INITIALIZE KEY SORT INPUT PHASE FOR INDEX GROUP 1 |
| BMC40813I | ESTIMATED NUMBER OF RECORDS = 6006 |
Figure 43  AFRPLAN with MSGLEVEL(2) (part 2 of 2)
Examples of RECOVER PLUS operations

This section describes and provides examples of some of the recovery scenarios that you can implement by using RECOVER PLUS.

The recovery scenarios show the different RECOVER PLUS execution phases. RECOVER PLUS analyzes all requests in the SYSIN data set and then constructs a plan that consists of the execution phases needed to accomplish the work requested. Table 13 describes the major execution phases.

Table 13  Major RECOVER PLUS execution phases

<table>
<thead>
<tr>
<th>Phase</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOG INPUT</td>
<td>The LOG INPUT phase reads the DB2 log records required for recovery and sends them to a sort task. If you specified MAXLOGS &gt; 1, RECOVER PLUS starts multiple subtasks to handle log file allocation. The main task processes log records. The subtasks handle allocation.</td>
</tr>
<tr>
<td>MERGE</td>
<td>The MERGE phase reads sorted log records and applies them to page images, from an image copy or, in the case of BACKOUT and LOGONLY, from the space itself. This phase recovers table spaces or indexes from logs, extracts keys and sends them to a sort, reads the space for BACKOUT or LOGONLY recovery, writes output image copy data sets, writes output change accumulation files, and performs many other tasks. Multiple MERGE phases can run in parallel (using the MAXLSORT option).</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>The UNLOAD phase extracts the index keys from a table space or partition that is not being recovered (and has no MERGE phase) and sends those keys to a sort task. A MERGE phase and an UNLOAD phase should not occur for the same table space or table space partition.</td>
</tr>
<tr>
<td>BUILD</td>
<td>The BUILD phase constructs indexes by using sorted keys extracted in a MERGE or UNLOAD phase.</td>
</tr>
<tr>
<td>SNAP</td>
<td>The SNAP phase uses XBM or SUF to perform a data set snap to restore an Instant Snapshot copy.</td>
</tr>
</tbody>
</table>

Multiple table space recovery and index rebuilds

Figure 44 shows the phases that are used when multiple table spaces are being recovered and the associated indexes are being rebuilt and output copies are requested. In this example, MAXLSORT is set to 1 so that multiple log sorts and parallel MERGE phases are not used. Log records are selected and sorted for all table spaces during the LOG INPUT phase and then are merged with the image copies of the table spaces during each MERGE phase. Index keys are also extracted during the MERGE phase and are passed directly to the sort routine. When the MERGE phase for a table space is complete, the BUILD phase constructs the indexes by using the
Parallel index key sorts and multitasking index rebuilds

sorted keys. The MERGE and BUILD phases repeat for each table space until all table spaces are recovered and all indexes are rebuilt. Sort tasks for log records and index keys are managed by the appropriate phases and depend on the options selected. Sort tasks are shown as shaded boxes in the background in Figure 44.

Figure 44 Phases used for table space recovery with indexes rebuilt

Parallel index key sorts and multitasking index rebuilds

Figure 45 on page 313 shows the phases that are used when you use the MAXKSORT option to specify that index key sorts are to run in parallel and that index rebuilds are multitasked. In this example, MAXLSORT is set to 1 so that multiple log sorts and parallel MERGE phases are not used and KSORTSHARE has no effect.
For a partitioned space, RECOVER PLUS can schedule a BUILD phase for each partition of a partitioning index after the completion of the MERGE phase for the corresponding table space partition. In this case, the BUILD phase for the index partition can overlap the MERGE phase for the next partition. The BUILD phase for the partitions of a partitioned index can also be scheduled after all MERGE phases for the table space partitions have completed.

RECOVER PLUS chooses the most efficient strategy to reduce sort times and the number of bytes that are sorted. The number of BUILD phases and index key sorts that can run concurrently is equal to the value of MAXKSORT.

Sort tasks are shown as shaded boxes in the background in Figure 45 on page 313.
Multiple log sorts, MERGE phases, and parallel index key sorts with multitasking index rebuilds

Figure 46 on page 315 shows the phases that are used when you use

- the MAXLSORT option to specify multiple log sorts and parallel MERGE phases
- the KSORTSHARE option to indicate that each MERGE phase has its own set of key sorts
- the MAXKSORT option to specify that index key sorts are to run in parallel and that index rebuilds are multitasked

RECOVER PLUS schedules concurrent log sorts and MERGE phases for the table spaces. The indexes for each table space are handled as in “Parallel index key sorts and multitasking index rebuilds” on page 312, except that the total number of BUILD phases and index key sorts that can run concurrently is equal to the value of MAXKSORT * MAXLSORT.

RECOVER PLUS chooses the most efficient strategy to reduce sort times and the number of bytes that are sorted.

Sort tasks are shown as shaded boxes in the background.
Multiple log sorts, MERGE phases, and parallel index key sorts with multitasking index rebuilds

Figure 46  Phases used for table space recovery with multiple log sorts and parallel index sorts with multitasking index rebuilds

OPTIONS MAXKSORT 2 MAXLSORT 2 KSORTSHARE NO
RECOVER TABLESPACE tableSpaceName1
TABLESPACE tableSpaceName1
REBUILD INDEX(ALL) TABLESPACE tableSpaceName1
REBUILD INDEX(ALL) TABLESPACE tableSpaceName1

(UTILINIT, ANALYZE, and UTILTERM phases also occur)
Figure 47 shows the phases that are used when table spaces and index spaces are being recovered. Log records are selected and sorted during the LOG INPUT phase and then are merged with the image copies of the spaces during the MERGE phases. The MERGE phase repeats for each table space and index. The sort task is shown as a shaded box in the background in Figure 47.

**Figure 47  Phases used for table space and index space recovery**

```
RECOVER TABLESPACE tableSpaceName1
TABLESPACE tableSpaceNamen
RECOVER INDEXSPACE(ALL) TABLESPACE tableSpaceName1
RECOVER INDEXSPACE(ALL) TABLESPACE tableSpaceNamen
```
**BACKOUT recovery**

Figure 48 shows the phases that are used when a table space and an index space are being recovered with the BACKOUT option to a specific point in time. Log records are selected and sorted during the LOG INPUT phase and then are merged with the table space and the index space. Image copies are not used with the BACKOUT strategy. In this case only active log records are needed. (Log records since the last quiesce are all that are required.)

**Figure 48  Phases used for BACKOUT recovery**

OPTIONS BACKOUT YES INDEXLOG YES
RECOVER TABLESPACE *tableSpaceName1* TOLOGPOINT LASTQUIESCE
RECOVER INDEXSPACE (*indexName1*)
Instant Snapshot recovery

Figure 49 shows the phases that are used when table spaces and index spaces are being recovered by using an Instant Snapshot copy. If the recovery is a TOCOPY request or if there is no log to apply and no output copies, the SNAP phase does not require the following LOG INPUT phase or MERGE phase. A MERGE phase is scheduled if keys are being extracted or output copies are being made.

Options XBMID DBXM
RECOVER TABLESPACE tableSpaceName1

TABLESPACE tableSpaceName
RECOVER INDEXSPACE(ALL) TABLESPACE tableSpaceName1

RECOVER INDEXSPACE(ALL) TABLESPACE tableSpaceName

Figure 49  Phases used for table space and index space recovery for Instant Snapshots
Examples of RECOVER PLUS jobs

This chapter presents the following topics:

Overview .......................................................... 320
Example 1: Previewing the recovery plan ................. 325
Example 2: Recovering to the current state .......... 334
    Example 2A: Using REBUILD INDEX ............... 334
    Example 2B: Using RECOVER INDEX ............... 340
Example 3: Creating and dynamically allocating output copies .... 359
Example 4: Recovering to a specified copy ............ 360
    Example 4A: Using TOCOPY with a named copy ... 360
    Example 4B: Using TOCOPY LASTCOPY .......... 360
Example 5: Recovering to a specific log point ....... 363
    Example 5A: Using the TOLOGPOINT option with RECOVER .... 363
    Example 5B: Using TOLOGPOINT and BACKOUT ... 364
Example 6: Extracting nonpartitioned index keys ... 365
Example 7: Building a nonpartitioned index ........... 366
Example 8: Recovering using tape-stacked data sets ... 367
    Example 8A: Making stacked image copies ....... 367
    Example 8B: Recovering with stacked image copies ... 368
Example 9: Recovering to a non-DB2 data set .......... 369
    Directing recovery output ......................... 369
Example 10: Overriding installation options .......... 372
    Example 10A: Overriding several installation options .... 372
    Example 10B: Overriding the IXRECP installation option .... 373
Example 11: Recovering using a change accumulation file .... 383
    Example 11A: Creating the change accumulation file .... 383
    Example 11B: Using the change accumulation file .... 383
Example 12: Using the MAINT parameter ................. 384
Example 13: Using the LOGSCAN command ............. 387
Example 14: Simulating recovery ....................... 390
Example 15: Using MAXKSORT for parallel index rebuilds ... 391
Example 16: Using MAXKSORT and recovering a table space .... 397
Example 17: Using timestamp recovery ............... 398
Example 18: Using the MAXLSORT default value ........ 399
Overview

This chapter provides example JCL for commonly-run RECOVER PLUS jobs. Example output is provided for some of the examples. For more information on the output produced by RECOVER PLUS jobs, see the following sections:

- “RECOVER PLUS data sets and RECOVER PLUS DD statements” on page 276
- “DD statements common to all RECOVER PLUS executions” on page 277
- “Using MSGLEVEL to control RECOVER PLUS output” on page 298

See Chapter 3, “RECOVER PLUS syntax,” for a description of the syntax options used in these examples.

Copies of the JCL for these examples are in members AFREXnn (where nn is the example number) in the HLQ.AFRSAMP installation data set (where HLQ represents the high-level qualifier specified during installation).

Table 14 provides cross references based on the utility parameters, commands, or keywords used in the examples. This table gives the cross references for the utility parameters first, followed by the commands, in the order they are most likely to be used, and their keywords in SYSIN.

**Table 14 Cross reference of examples by command or keyword (part 1 of 5)**

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
<th>Page number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utility parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PARM VALUE</td>
<td>all examples</td>
<td>325 through 398</td>
</tr>
<tr>
<td>REGION</td>
<td>all examples</td>
<td>325 through 398</td>
</tr>
<tr>
<td>NEW/RESTART</td>
<td>“Example 1: Previewing the recovery plan”</td>
<td>325</td>
</tr>
<tr>
<td></td>
<td>“Example 2B: Using RECOVER INDEX”</td>
<td>340</td>
</tr>
<tr>
<td></td>
<td>“Example 3: Creating and dynamically allocating output copies” through “Example 9: Recovering to a non-DB2 data set”</td>
<td>359 through 369</td>
</tr>
<tr>
<td></td>
<td>“Example 10B: Overriding the IXRECP installation option”</td>
<td>373</td>
</tr>
<tr>
<td></td>
<td>“Example 11: Recovering using a change accumulation file”</td>
<td>383</td>
</tr>
<tr>
<td></td>
<td>“Example 13: Using the LOGSCAN command” through “Example 17: Using timestamp recovery”</td>
<td>387 through 398</td>
</tr>
</tbody>
</table>
Table 14  Cross reference of examples by command or keyword (part 2 of 5)

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
<th>Page number</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Utility parameters, continued</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NEW</td>
<td>“Example 2A: Using REBUILD INDEX”</td>
<td>334</td>
</tr>
<tr>
<td></td>
<td>“Example 10A: Overriding several installation options”</td>
<td>372</td>
</tr>
<tr>
<td>MSGLEVEL</td>
<td>“Example 2: Recovering to the current state” through “Example 9: Recovering to a non-DB2 data set”</td>
<td>325 through 369</td>
</tr>
<tr>
<td></td>
<td>“Example 10B: Overriding the IXRECP installation option” through “Example 17: Using timestamp recovery”</td>
<td>373 through 398</td>
</tr>
<tr>
<td>MAINT</td>
<td>“Example 12: Using the MAINT parameter”</td>
<td>384</td>
</tr>
<tr>
<td><strong>Commands and keywords</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>OPTIONS command</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTIONS</td>
<td>“Example 2: Recovering to the current state”</td>
<td>334</td>
</tr>
<tr>
<td></td>
<td>“Example 5: Recovering to a specific log point”</td>
<td>363</td>
</tr>
<tr>
<td>BACKOUT</td>
<td>“Example 17: Using timestamp recovery”</td>
<td>398</td>
</tr>
<tr>
<td>EARLYRECALL</td>
<td>“Example 2: Recovering to the current state”</td>
<td>334</td>
</tr>
<tr>
<td>INDEXLOG</td>
<td>“Example 2: Recovering to the current state”</td>
<td>334</td>
</tr>
<tr>
<td></td>
<td>“Example 5: Recovering to a specific log point”</td>
<td>363</td>
</tr>
<tr>
<td></td>
<td>“Example 13: Using the LOGSCAN command”</td>
<td>387</td>
</tr>
<tr>
<td>IXRECP</td>
<td>“Example 10: Overriding installation options”</td>
<td>372</td>
</tr>
<tr>
<td>MAXKSORT</td>
<td>“Example 15: Using MAXKSORT for parallel index rebuilds”</td>
<td>391</td>
</tr>
<tr>
<td></td>
<td>“Example 16: Using MAXKSORT and recovering a table space”</td>
<td>397</td>
</tr>
<tr>
<td>MAXLSORT</td>
<td>“Example 18: Using the MAXLSORT default value”</td>
<td>399</td>
</tr>
<tr>
<td>RECOVERYPOINT</td>
<td>“Example 17: Using timestamp recovery”</td>
<td>398</td>
</tr>
<tr>
<td>RESOURCE SELECTION</td>
<td>“Example 10: Overriding installation options”</td>
<td>372</td>
</tr>
<tr>
<td>SIMULATE YES</td>
<td>“Example 14: Simulating recovery”</td>
<td>390</td>
</tr>
<tr>
<td>SORTDEVT</td>
<td>“Example 10: Overriding installation options”</td>
<td>372</td>
</tr>
<tr>
<td>SORTNUM</td>
<td>“Example 10: Overriding installation options”</td>
<td>372</td>
</tr>
<tr>
<td>USEACCUM</td>
<td>“Example 11: Recovering using a change accumulation file”</td>
<td>383</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>“Example 17: Using timestamp recovery”</td>
<td>398</td>
</tr>
<tr>
<td><strong>OUTPUT command</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUTPUT</td>
<td>“Example 3: Creating and dynamically allocating output copies”</td>
<td>359</td>
</tr>
<tr>
<td></td>
<td>“Example 17: Using timestamp recovery”</td>
<td>398</td>
</tr>
<tr>
<td>DSNAME</td>
<td>“Example 3: Creating and dynamically allocating output copies”</td>
<td>359</td>
</tr>
<tr>
<td></td>
<td>“Example 17: Using timestamp recovery”</td>
<td>398</td>
</tr>
</tbody>
</table>
### Table 14  Cross reference of examples by command or keyword (part 3 of 5)

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
<th>Page number</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Commands and keywords</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>OUTPUT command, continued</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MODELDCB</td>
<td>“Example 3: Creating and dynamically allocating output copies”</td>
<td>359</td>
</tr>
<tr>
<td>SPACE</td>
<td>“Example 3: Creating and dynamically allocating output copies”</td>
<td>359</td>
</tr>
<tr>
<td>UNIT</td>
<td>“Example 3: Creating and dynamically allocating output copies”</td>
<td>359</td>
</tr>
<tr>
<td>MODELDCB</td>
<td>“Example 3: Creating and dynamically allocating output copies”</td>
<td>359</td>
</tr>
<tr>
<td>SPACE</td>
<td>“Example 3: Creating and dynamically allocating output copies”</td>
<td>359</td>
</tr>
<tr>
<td>UNIT</td>
<td>“Example 3: Creating and dynamically allocating output copies”</td>
<td>359</td>
</tr>
<tr>
<td><strong>RECOVER command</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECOVER</td>
<td>“Example 1: Previewing the recovery plan” through “Example 6: Extracting nonpartitioned index keys”</td>
<td>325 through 365</td>
</tr>
<tr>
<td></td>
<td>“Example 8: Recovering using tape-stacked data sets” through “Example 11: Recovering using a change accumulation file”</td>
<td>367 through 383</td>
</tr>
<tr>
<td></td>
<td>“Example 16: Using MAXKSORT and recovering a table space” through “Example 17: Using timestamp recovery”</td>
<td>397 through 398</td>
</tr>
<tr>
<td>ANALYZE</td>
<td>“Example 1: Previewing the recovery plan”</td>
<td>325</td>
</tr>
<tr>
<td>BACKOUT</td>
<td>“Example 5: Recovering to a specific log point”</td>
<td>363</td>
</tr>
<tr>
<td>DSNUM</td>
<td>“Example 6: Extracting nonpartitioned index keys”</td>
<td>365</td>
</tr>
<tr>
<td>INDEP OUTSPACE</td>
<td>“Example 9: Recovering to a non-DB2 data set”</td>
<td>369</td>
</tr>
<tr>
<td>INDEX</td>
<td>“Example 5: Recovering to a specific log point”</td>
<td>363</td>
</tr>
<tr>
<td></td>
<td>“Example 17: Using timestamp recovery”</td>
<td>398</td>
</tr>
<tr>
<td>LASTCOPY</td>
<td>“Example 4: Recovering to a specified copy”</td>
<td>360</td>
</tr>
<tr>
<td></td>
<td>“Example 8: Recovering using tape-stacked data sets”</td>
<td>367</td>
</tr>
<tr>
<td></td>
<td>“Example 10: Overriding installation options”</td>
<td>372</td>
</tr>
<tr>
<td>LASTQUIESCE</td>
<td>“Example 5: Recovering to a specific log point”</td>
<td>363</td>
</tr>
<tr>
<td>OUTCOPY</td>
<td>“Example 3: Creating and dynamically allocating output copies”</td>
<td>359</td>
</tr>
<tr>
<td>OUTCOPYDDN</td>
<td>“Example 3: Creating and dynamically allocating output copies”</td>
<td>359</td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>“Example 1: Previewing the recovery plan” through “Example 6: Extracting nonpartitioned index keys”</td>
<td>325 through 365</td>
</tr>
<tr>
<td></td>
<td>“Example 8: Recovering using tape-stacked data sets” through “Example 11: Recovering using a change accumulation file”</td>
<td>367 through 383</td>
</tr>
<tr>
<td></td>
<td>“Example 16: Using MAXKSORT and recovering a table space”</td>
<td>397</td>
</tr>
<tr>
<td>Command or keyword</td>
<td>Relevant examples</td>
<td>Page number</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Commands and keywords</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECOVER command, continued</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOCOPY</td>
<td>“Example 4: Recovering to a specified copy”</td>
<td>360</td>
</tr>
<tr>
<td></td>
<td>“Example 8: Recovering using tape-stacked data sets”</td>
<td>367</td>
</tr>
<tr>
<td></td>
<td>“Example 10: Overriding installation options”</td>
<td>372</td>
</tr>
<tr>
<td>TOLOGPOINT</td>
<td>“Example 5: Recovering to a specific log point” on page 363</td>
<td>363</td>
</tr>
<tr>
<td>REBUILD command</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REBUILD</td>
<td>“Example 2: Recovering to the current state”</td>
<td>334</td>
</tr>
<tr>
<td></td>
<td>“Example 4: Recovering to a specified copy”</td>
<td>360</td>
</tr>
<tr>
<td></td>
<td>“Example 6: Extracting nonpartitioned index keys”</td>
<td>365</td>
</tr>
<tr>
<td></td>
<td>“Example 9: Recovering to a non-DB2 data set”</td>
<td>369</td>
</tr>
<tr>
<td></td>
<td>“Example 11: Recovering using a change accumulation file”</td>
<td>383</td>
</tr>
<tr>
<td></td>
<td>“Example 15: Using MAXKSORT for parallel index rebuilds”</td>
<td>391</td>
</tr>
<tr>
<td></td>
<td>“Example 16: Using MAXKSORT and recovering a table space”</td>
<td>397</td>
</tr>
<tr>
<td>INDEP OUTSPACE</td>
<td>“Example 9: Recovering to a non-DB2 data set”</td>
<td>369</td>
</tr>
<tr>
<td>INDEX</td>
<td>“Example 2: Recovering to the current state”</td>
<td>334</td>
</tr>
<tr>
<td></td>
<td>“Example 4: Recovering to a specified copy”</td>
<td>360</td>
</tr>
<tr>
<td></td>
<td>“Example 6: Extracting nonpartitioned index keys”</td>
<td>365</td>
</tr>
<tr>
<td></td>
<td>“Example 9: Recovering to a non-DB2 data set”</td>
<td>369</td>
</tr>
<tr>
<td></td>
<td>“Example 11: Recovering using a change accumulation file”</td>
<td>383</td>
</tr>
<tr>
<td></td>
<td>“Example 15: Using MAXKSORT for parallel index rebuilds”</td>
<td>391</td>
</tr>
<tr>
<td></td>
<td>“Example 16: Using MAXKSORT and recovering a table space”</td>
<td>397</td>
</tr>
<tr>
<td>NOWORKDDN</td>
<td>“Example 2: Recovering to the current state”</td>
<td>334</td>
</tr>
<tr>
<td></td>
<td>“Example 4: Recovering to a specified copy”</td>
<td>360</td>
</tr>
<tr>
<td></td>
<td>“Example 9: Recovering to a non-DB2 data set”</td>
<td>369</td>
</tr>
<tr>
<td></td>
<td>“Example 11: Recovering using a change accumulation file”</td>
<td>383</td>
</tr>
<tr>
<td>PART</td>
<td>“Example 6: Extracting nonpartitioned index keys”</td>
<td>365</td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>“Example 4: Recovering to a specified copy”</td>
<td>360</td>
</tr>
<tr>
<td></td>
<td>“Example 15: Using MAXKSORT for parallel index rebuilds”</td>
<td>391</td>
</tr>
<tr>
<td></td>
<td>“Example 16: Using MAXKSORT and recovering a table space”</td>
<td>397</td>
</tr>
<tr>
<td>RECOVER UNLOADKEYS command</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECOVER UNLOADKEYS</td>
<td>“Example 6: Extracting nonpartitioned index keys”</td>
<td>365</td>
</tr>
<tr>
<td>PART</td>
<td>“Example 6: Extracting nonpartitioned index keys”</td>
<td>365</td>
</tr>
<tr>
<td>RECOVER BUILDINDEX command</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECOVER BUILDINDEX</td>
<td>“Example 7: Building a nonpartitioned index”</td>
<td>366</td>
</tr>
</tbody>
</table>
### Table 14  Cross reference of examples by command or keyword (part 5 of 5)

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
<th>Page number</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Commands and keywords</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACCUM command</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACCUM</td>
<td>“Example 11: Recovering using a change accumulation file”</td>
<td>383</td>
</tr>
<tr>
<td>GROUP</td>
<td>“Example 11: Recovering using a change accumulation file”</td>
<td>383</td>
</tr>
<tr>
<td><strong>LOGSCAN Command</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOGSCAN</td>
<td>“Example 13: Using the LOGSCAN command”</td>
<td>387</td>
</tr>
<tr>
<td>BACKOUT</td>
<td>“Example 13: Using the LOGSCAN command”</td>
<td>387</td>
</tr>
<tr>
<td>INDEXSPACE</td>
<td>“Example 13: Using the LOGSCAN command”</td>
<td>387</td>
</tr>
<tr>
<td>LASTQUIESCE</td>
<td>“Example 13: Using the LOGSCAN command”</td>
<td>387</td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>“Example 13: Using the LOGSCAN command”</td>
<td>387</td>
</tr>
<tr>
<td>TOLOGPOINT</td>
<td>“Example 13: Using the LOGSCAN command”</td>
<td>387</td>
</tr>
<tr>
<td><strong>SIMRCVR command</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMRCVR</td>
<td>“Example 14: Simulating recovery”</td>
<td>390</td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>“Example 14: Simulating recovery”</td>
<td>390</td>
</tr>
<tr>
<td><strong>SIMRBLD command</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMRBLD</td>
<td>“Example 14: Simulating recovery”</td>
<td>390</td>
</tr>
<tr>
<td>INDEX</td>
<td>“Example 14: Simulating recovery”</td>
<td>390</td>
</tr>
</tbody>
</table>
Example 1: Previewing the recovery plan

This example illustrates the use of the ANALYZE ONLY option with RECOVER TABLESPACE to obtain a preview of the recovery plan. When you use the ANALYZE ONLY option, RECOVER PLUS determines the data sets required for recovery and produces a recovery plan. This example also uses the SYSPICK DD statement to list all of the tape and cartridge volumes that will be dynamically allocated during recovery. The JCL for this example is in Figure 50.

Figure 50  Example 1—JCL using ANALYZE ONLY

```plaintext
//AFREX01 JOB (PAFR),'EXAMPLE 1',
  //         CLASS=Q,
  //       MSGCLASS=X,NOTIFY=&SYSUID
      /*ROUTE XEQ BMCPXL1
      /*JOBPARM SYSAFF=DB2A
      /*
      //RECOVER EXEC PGM=AFRMAIN,REGION=0M,
        //         PARM='DGE,EXAMPLE01,NEW/RESTART'
        //STEPLIB DD DISP=SHR,DSN=product.libraries
        //         DD DISP=SHR,DSN=DB2.DSNEXIT
        //         DD DISP=SHR,DSN=DB2.DSNLOAD
        //SYSIN     DD *
        RECOVER TABLESPACE BMCDBSMP.BMCTS001
          LOGSORT ANALYZE ONLY
        /*
        //SYSPICK   DD SYSOUT=*
        //
```

Figure 51 through Figure 55 on page 333 show the following output files from this RECOVER PLUS job:

- AFRSUMRY
- AFRSTMT
- AFROSUM
- AFRPLAN
- SYSPICK

See Table 10 on page 277 for a description of each of these files.

Figure 51  Example 1—AFRSUMRY using ANALYZE ONLY (part 1 of 2)
Example 1: Previewing the recovery plan

Figure 51 Example 1—AFRSTMT using ANALYZE ONLY (part 2 of 3)

BMC40001I BMC9601I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276
BMC40876I MAINT: BPJ0279 BPJ0288

| BMC40002I UTILITY ID = RDAJBM.AFR01. DB2 SUBSYSTEM ID = DXW |
| BMC40004I ATTEMPTING TO CONNECT TO DB2 SUBSYSTEM DXW, USING PLAN AFRB101T |
| BMC40024I A GROUP ATTACH WAS PERFORMED WITH DXW |
| BMC40067I DB2 DXW IS MEMBER DXW4 IN DATA SHARING GROUP DSNDXW |

BMC40475I INPUT STATEMENTS PRINTED IN AFRSTMT

BMC40336I ANALYZE FINISHED.
BMC40288I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40336I PLANNING FINISHED.
BMC40288I ELAPSED TIME = 00:00:03, UTILITY ELAPSED TIME = 00:00:05
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

| BMC40005I UTILITY EXECUTION COMPLETE 01/05/2011 09:00:42, RETURN CODE = 0 |

Figure 52 Example 1—AFRSUMRY using ANALYZE ONLY (part 1 of 3)

BMC40018I BMC9601I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC40673I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40937I z/OS RELEASE = 1.9.00, PID = HBB7740, SMS RELEASE = 1.9.0
BMC40157I AVAILABLE REGION BELOW 16M = 8836K, AVAILABLE REGION ABOVE 16M = 1574392K, NUMBER OF CPUS = 3
BMC40931I PARM LIST: DB2 SSID = DXW
BMC40930I PARM NOT CODED - UTILID
BMC40930I PARM NOT CODED - CHECKPT
BMC40930I PARM NOT CODED - RDB2STAT
BMC40930I PARM NOT CODED - AFROPTS
BMC40930I PARM NOT CODED - ACAOPTS

BMC40924I OPTIONS: PLANRECV = AFRB101T (MAIN PLAN)
BMC40925I TBUFFS = 1000 (BUFFER MANAGER PAGES)
BMC40944I OPNDB2ID = YES (ACQUIRE RACF AUTHORITY OF DB2)
BMC40926I CHECKPT = PHASE (CHECKPOINTING)
BMC40945I CHECKINT = 0 (CHECKPOINT INTERVAL MINUTES)
BMC40927I RDB2STAT = YES (RESET DB2 OBJECT STATUS)
BMC40946I AMSCAT = NO (USE ‘CATALOG’ IN IDCAMS INPUT)
BMC40928I SMCORE = (OK, OK) (SORT CORE VALUES)
BMC40931I RESINV = (OK) (MEMORY BELOW 16MB EXCLUDED FROM USE BY SORT)
BMC40938I WKUNIT = SYSALLDA (WORK UNIT NAME)
BMC40956I RCLTSK = 10 (MAXIMUM RECALL TASKS)
BMC40941I IXRECP = NO (SET INDEX RECOVER PENDING)
BMC40911I MAXDRIVES = 0 (DEFAULT MAX TAPE DRIVES)
BMC409892I MAXLOGS = 3 (DEFAULT MAX CONCURRENT LOG FILES)
BMC40942I MAXSORT = 6 (DEFAULT) (MAX CONCURRENT INDEX SORTS)
BMC40630I KSORTSHARE = YES (KEY SORTS SHARED BY ALL TASKS)
BMC406192I MAXSORT = 50 (MAX CONCURRENT LOG SORTS)
BMC40962I SORTDEV1 = SORT DEFAULT (DEVICE TYPE FOR LOG AND INDEX SORTS)
BMC40963I SORTNUM = SORT DEFAULT (WORK DATASETS FOR LOG AND INDEX SORTS)
BMC40964I OUTCOPY = ASCODED (OUTCOPY CREATION DIRECTIVE)
BMC40547I INDEXLOG = NO (DEFAULT USE OF INDEXLOG STRATEGY)
BMC40878I WTOR = YES (ISSUE WTOR IF SPACE REMAINS IN STOPP STATUS)
BMC40537I ERRCONT = 10 (DEFAULT MAX SEVERE ERRORS)
BMC40241I AUTOSIZE = YES (AUTO SIZE ACCUM OUTPUT FILES)
BMC40241I PUBLICPLAN = YES (GRANT PLAN TO PUBLIC)
### Example 1—AFRSTMT using ANALYZE ONLY (part 2 of 3)

| BMC40408I | BINDQUALIFIER = BMCAFR (QUALIFIER USED FOR DYNAMIC BIND) |
| BMC96233I | BACKOUT = AUTO (DEFAULT POINT-IN-TIME RECOVERY STRATEGY) |
| BMC96235I | AUX = NO (INCLUDE XML AND/OR LOB AUXILIARY OBJECTS) |
| BMC40967I | TAPE DEVICE01 = 3590-1 |
| BMC40967I | TAPE DEVICE02 = 3490 |
| BMC40967I | TAPE DEVICE03 = 3480X |
| BMC40967I | TAPE DEVICE04 = ATLVT |
| BMC40967I | TAPE DEVICE05 = ATL3592 |
| BMC40967I | TAPE DEVICE06 = CART |
| BMC40967I | TAPE DEVICE07 = CARTVT |
| BMC40967I | TAPE DEVICE08 = CRT8NS |
| BMC40967I | TAPE DEVICE09 = CRT9NS |
| BMC40967I | TAPE DEVICE10 = DANBARR |
| BMC40967I | TAPE DEVICE11 = EXT3592 |
| BMC40967I | TAPE DEVICE12 = IBMATL |
| BMC40967I | TAPE DEVICE13 = IBMVT |
| BMC40967I | TAPE DEVICE14 = SA3590 |
| BMC40967I | TAPE DEVICE15 = TAPEABL |
| BMC40967I | TAPE DEVICE16 = 3590B |
| BMC40967I | TAPE DEVICE17 = 3590E |
| BMC40967I | TAPE DEVICE18 = 3480 |
| BMC40967I | TAPE DEVICE19 = 3400-9 |
| BMC40967I | TAPE DEVICE20 = SYS3480R |
| BMC40967I | TAPE DEVICE21 = SYS348XR |
| BMC40287I | ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00 |
| BMC40288I | CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00 |
| BMC40954I | AVAILABLE MEMORY BELOW THE LINE = 8744K, TOTAL AVAILABLE MEMORY (AT LEAST) = 1582696K |
| BMC40335I | RECOVER PLUS FOR DB2 V10.1.00 |
| BMC40876I | MAINT: NO RECOVER PLUS PTFS APPLIED |
| BMC96010I | SOLUTION COMMON CODE V10.1.00 |
| BMC40876I | MAINT: NO SOLUTION COMMON CODE PTFS APPLIED |
| BMC40335I | ANALYZE STARTING 01/05/2011 09:00:38 |
| BMC40335I | SCAN SYSTB FINISHED. |
| BMC40335I | SCAN SYSIX FINISHED. |
| BMC40954I | BYT:0, 0, 0; BLK:0, 0, 0; TOT:0, 0, 0; iter: 17; RC:8 |
| BMC40611I | INPUT STATEMENTS: |
| BMC40101I | RECOVER TABLESPACE AFREX01.TSEX1P1 |
| BMC40101I | LOGSORT ANALYZE ONLY |
| BMC40101I | BMC40335I ANALYZE STARTING 01/05/2011 09:00:38 |
| BMC40335I | CATALOG LOOKUP STARTING 01/05/2011 09:00:38 |
| BMC40335I | PROC RO STMT FINISHED. |
| BMC40336I | SCAN SYSTB FINISHED. |
| BMC40336I | SCAN SYSIX FINISHED. |

**Figure 52** Example 1—AFRSTMT using ANALYZE ONLY (part 2 of 3)
### Figure 52  Example 1—AFRSTMT using ANALYZE ONLY (part 3 of 3)

<table>
<thead>
<tr>
<th>BMC40287I</th>
<th>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
</tbody>
</table>

BMC40336I TS LOOKUP FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40633I USING STANDARD DB2 SECURITY

BMC40336I DB LOOKUP FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40336I TB LOOKUP FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40336I SG LOOKUP FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40336I SCAN SYSVOL FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40336I CHECK DS FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40336I HDR PG RD FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40336I SYSCOPY LOOKUP FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40336I CATALOG LOOKUP FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40335I SYNCHRONIZATION STARTING 01/05/2011 09:00:38
BMC40336I SYNCHRONIZATION FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC96306I MAXLSORT REDUCED FROM 50 TO 24 DUE TO MEMORY CONSTRAINTS
BMC40336I DSCAN FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40335I PLANNING STARTING 01/05/2011 09:00:39
BMC40459I BSDS DSNDWX.DWX4.BSDS01 SELECTED
BMC40066I TIME TO READ SYSLGRNX WAS 00:00:00

### Figure 53  Example 1—AFROSUM using ANALYZE ONLY (part 1 of 3)

<table>
<thead>
<tr>
<th>BMC40018I</th>
<th>RECOVER PLUS FOR DB2 V10.1.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40473I</td>
<td>COPYRIGHT BMC SOFTWARE INC. 1991-2011</td>
</tr>
<tr>
<td>BMC40173I</td>
<td>RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884</td>
</tr>
<tr>
<td>BMC40474I</td>
<td>CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO <a href="mailto:SUPPORT@BMC.COM">SUPPORT@BMC.COM</a></td>
</tr>
</tbody>
</table>
**Figure 53  Example 1—AFROSUM using ANALYZE ONLY (part 2 of 3)**

<table>
<thead>
<tr>
<th>BMC407091</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 1 OF 24</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = X'0231'  PSID = X'000E'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = SYSDEFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40820I</th>
<th>TABLESPACE RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'50B146760692' TO LOGPOINT = X'50B147AAADD'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40840I</th>
<th>INPUT COPY / SYSCOPY DATA:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40585I</td>
<td>2011-01-05 09.00.22.7428 RBA/LRSN X'50B14670692' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = ACP.QA.AFREX01.TSEX1P1.D00.T090018</td>
</tr>
<tr>
<td>BMC40641I</td>
<td>SHRLEVEL = REFERENCE SITETYPE = LP DSNUM = 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC407091</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 2 OF 24</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = X'0231'  PSID = X'000E'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = SYSDEFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40820I</th>
<th>TABLESPACE RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'50B146760692' TO LOGPOINT = X'50B147AAADD'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40840I</th>
<th>INPUT COPY / SYSCOPY DATA:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40585I</td>
<td>2011-01-05 09.00.22.7428 RBA/LRSN X'50B14670692' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = ACP.QA.AFREX01.TSEX1P1.D00.T090018</td>
</tr>
<tr>
<td>BMC40641I</td>
<td>SHRLEVEL = REFERENCE SITETYPE = LP DSNUM = 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC407091</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 3 OF 24</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = X'0231'  PSID = X'000E'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = SYSDEFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40820I</th>
<th>TABLESPACE RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'50B146760692' TO LOGPOINT = X'50B147AAADD'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40840I</th>
<th>INPUT COPY / SYSCOPY DATA:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40585I</td>
<td>2011-01-05 09.00.22.7428 RBA/LRSN X'50B14670692' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = ACP.QA.AFREX01.TSEX1P1.D00.T090018</td>
</tr>
<tr>
<td>BMC40641I</td>
<td>SHRLEVEL = REFERENCE SITETYPE = LP DSNUM = 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC407091</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 23 OF 24</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = X'0231'  PSID = X'000E'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = SYSDEFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>
### Figure 53  Example 1—AFROSUM using ANALYZE ONLY (part 3 of 3)

<table>
<thead>
<tr>
<th>BMC40820I</th>
<th>TABLESPACE RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'C50B14670692' TO LOGPOINT = X'C50B147AAADD'</td>
</tr>
<tr>
<td>BMC40840I</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>2011-01-05 09.00.22.7428 RBA/LRSN X'C50B14670692' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'C50B14670692' TO LOGPOINT = X'C50B147AAADD'</td>
</tr>
<tr>
<td>BMC40840I</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>2011-01-05 09.00.22.7428 RBA/LRSN X'C50B14670692' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'C50B14670692' TO LOGPOINT = X'C50B147AAADD'</td>
</tr>
</tbody>
</table>

### Figure 54  Example 1—AFRPLAN using ANALYZE ONLY (part 1 of 4)

<table>
<thead>
<tr>
<th>BMC40018I</th>
<th>RECOVER PLUS FOR DB2 V10.1.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40473I</td>
<td>COPYRIGHT BMC SOFTWARE INC. 1991-2011</td>
</tr>
<tr>
<td>BMC40673I</td>
<td>RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884</td>
</tr>
<tr>
<td>BMC40474I</td>
<td>CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO <a href="mailto:SUPPORT@BMC.COM">SUPPORT@BMC.COM</a></td>
</tr>
<tr>
<td>BMC40755I</td>
<td>ANALYSIS PLAN SUMMARY:</td>
</tr>
<tr>
<td>BMC40701I</td>
<td>PHASE: MERGE</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>PHASE INITIALIZATION:</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFXRO1.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE INPUT IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = ACP.QA.AFXRO1.TSEX1P1.D00.T090018</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>ALLOCATE 2 INPUT FILE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>OPEN IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = ACP.QA.AFXRO1.TSEX1P1.D00.T090018</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFXRO1.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFXRO1.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
</tbody>
</table>
Figure 54  Example 1—AFRPLAN using ANALYZE ONLY (part 2 of 4)

<table>
<thead>
<tr>
<th>BMC40702I</th>
<th>STEP: MERGE SPACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40351I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40358I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40702I</th>
<th>STEP: MERGE SPACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40702I</th>
<th>STEP: MERGE SPACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC4038I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40702I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>

Chapter 5  Examples of RECOVER PLUS jobs  331
### Figure 54  Example 1—AFRPLAN using ANALYZE ONLY  (part 3 of 4)

<table>
<thead>
<tr>
<th>BMC40738I</th>
<th>MERGE SPACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>

```
BMC40701I PHASE:  MERGE
BMC40721I PHASE INITIALIZATION:
BMC40301I  TABLESPACE = AFREX01.TSEX1P1
BMC40305I  DSNUM = 23
BMC40738I  SET RECOVERY PENDING (RECP) FLAG
BMC40738I  CREATE SPACE
BMC40345I  SPACE = AFREX01.TSEX1P1
BMC40305I  DSNUM = 23
BMC40738I  ALLOCATE SPACE (READ/WRITE)
BMC40345I  SPACE = AFREX01.TSEX1P1
BMC40305I  DSNUM = 23
BMC40738I  CONNECT TO SPACE
BMC40738I  ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I  FILE OPTIMIZATION
BMC40738I  ALLOCATE 2 SPACE OUTPUT BUFFER(S)
BMC40738I  ALLOCATE SPACEMAP BUFFER (4096 BYTES)
BMC40738I  ALLOCATE PASTE BUFFER (4096 BYTES)
BMC40702I  STEP:  MERGE SPACE
BMC40738I  MERGE SPACE
BMC40301I  TABLESPACE = AFREX01.TSEX1P1
BMC40305I  DSNUM = 23
BMC40735I  PHASE TERMINATION:
BMC40738I  DISCONNECT FROM SPACE
BMC40738I  DEALLOCATE SPACE
BMC40345I  SPACE = AFREX01.TSEX1P1
BMC40305I  DSNUM = 23
BMC40738I  FREE PAGE CHECK BUFFER
BMC40738I  FREE SPACEMAP BUFFER
BMC40738I  FREE PASTE BUFFER
BMC40738I  FREE 2 OUTPUT SPACE BUFFER(S)
BMC40738I  RESET RECOVERY PENDING (RECP) FLAG
BMC40738I  RECORD RESTART INFORMATION
```

BMC40701I PHASE:  MERGE
BMC40721I PHASE INITIALIZATION:
BMC40301I  TABLESPACE = AFREX01.TSEX1P1
BMC40305I  DSNUM = 24
BMC40738I  SET RECOVERY PENDING (RECP) FLAG
BMC40738I  CREATE SPACE
BMC40345I  SPACE = AFREX01.TSEX1P1
BMC40305I  DSNUM = 24
BMC40738I  ALLOCATE SPACE (READ/WRITE)
BMC40345I  SPACE = AFREX01.TSEX1P1
BMC40305I  DSNUM = 24
BMC40738I  CONNECT TO SPACE
BMC40738I  ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I  FILE OPTIMIZATION
BMC40738I  ALLOCATE 2 SPACE OUTPUT BUFFER(S)
BMC40738I  ALLOCATE SPACEMAP BUFFER (4096 BYTES)
### Figure 54  Example 1—AFRPLAN using ANALYZE ONLY (part 4 of 4)

<table>
<thead>
<tr>
<th>BMC40738I</th>
<th>ALLOCATE PASTE BUFFER (4096 BYTES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40702I</td>
<td>STEP: MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CLOSE IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = ACP.QA.AFREX01.TSEX1P1.D00.T090018</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 INPUT FILE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE INPUT IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = ACP.QA.AFREX01.TSEX1P1.D00.T090018</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>

### Figure 55  Example 1—SYSPICK using ANALYZE ONLY

<table>
<thead>
<tr>
<th>DATE 01/05/2011 RECOVER PLUS FOR DB2</th>
<th>VOLUME 001 of 001</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME 09:00:42</td>
<td>VOLUME PICK LIST</td>
</tr>
<tr>
<td>BY VOLUME SERIAL NUMBER</td>
<td></td>
</tr>
<tr>
<td>VOLSER 159495</td>
<td>CART</td>
</tr>
<tr>
<td>DEV TYPE ACP.QA.AFREX01.TSEX1P1.D00.T090018</td>
<td>SEQ 001</td>
</tr>
<tr>
<td>DATASET NAME</td>
<td></td>
</tr>
<tr>
<td>SEQ 001</td>
<td></td>
</tr>
<tr>
<td>DATE 01/05/2011 RECOVER PLUS FOR DB2</td>
<td>PAGE 001 of 001</td>
</tr>
<tr>
<td>TIME 09:00:42</td>
<td>VOLUME PICK LIST</td>
</tr>
<tr>
<td>BY ORDER OF USAGE</td>
<td></td>
</tr>
<tr>
<td>VOLSER 159495</td>
<td>CART</td>
</tr>
<tr>
<td>DEV TYPE ACP.QA.AFREX01.TSEX1P1.D00.T090018</td>
<td>SEQ 001</td>
</tr>
<tr>
<td>DATASET NAME</td>
<td></td>
</tr>
<tr>
<td>SEQ 001</td>
<td></td>
</tr>
</tbody>
</table>
Example 2: Recovering to the current state

This example illustrates two ways of recovering a partitioned table space and an index on the table in the table space to the current state. One job uses REBUILD INDEX (Figure 56), and the other job uses RECOVER INDEX (Figure 60 on page 340). The EARLYRECALL option is also used in both jobs, causing data sets to be scheduled for recall before other activities begin.

Example 2A: Using REBUILD INDEX

In the first job in this example, the index is rebuilt from the rows in this recovered image. Note the use of the NOWORKDDN option to send the extracted index keys directly to the sort routine. Also, note the use of MSGLEVEL(0), which produces less output than the default of MSGLEVEL(1).

Figure 56  Example 2A—JCL for a simple RECOVER TABLESPACE and REBUILD INDEX

Example 2A—JCL for a simple RECOVER TABLESPACE and REBUILD INDEX

Figure 57 on page 335 through Figure 59 on page 337 show the output produced by this JCL, which includes the following files:

- AFRSUMRY
- AFRSTMT
- AFRPRINT

See Table 10 on page 277 for a description of each of these files.
Example 2A: Using REBUILD INDEX

Figure 57 Example 2A—AFRSUMRY for a simple REBUILD INDEX

```
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40011I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40601I SOLUTION COMMON CODE V10.1.00
BMC40876I UTILITY EXECUTION COMPLETE 01/13/2011 08:35:56, RETURN CODE = 4
BMC40876I MAINT: NO RECOVER PLUS PTFs APPLIED
BMC96010I SOLUTION COMMON CODE V10.1.00
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFs APPLIED
BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276
BMC40876I MAINT: BPJ0279 BPJ0288
BMC40001I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
BMC40001I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40601I SOLUTION COMMON CODE V10.1.00
BMC40876I UTILITY EXECUTION COMPLETE 01/13/2011 08:35:56, RETURN CODE = 4
BMC40876I MAINT: NO RECOVER PLUS PTFs APPLIED
BMC96010I SOLUTION COMMON CODE V10.1.00
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFs APPLIED
BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276
BMC40876I MAINT: BPJ0279 BPJ0288
BMC40001I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
```

Figure 58 Example 2A—AFRSTMT for a simple REBUILD INDEX (part 1 of 3)

```
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40011I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40601I SOLUTION COMMON CODE V10.1.00
BMC40876I UTILITY EXECUTION COMPLETE 01/13/2011 08:35:56, RETURN CODE = 4
BMC40876I MAINT: NO RECOVER PLUS PTFs APPLIED
BMC96010I SOLUTION COMMON CODE V10.1.00
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFs APPLIED
BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276
BMC40876I MAINT: BPJ0279 BPJ0288
BMC40001I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
BMC40001I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40601I SOLUTION COMMON CODE V10.1.00
BMC40876I UTILITY EXECUTION COMPLETE 01/13/2011 08:35:56, RETURN CODE = 4
BMC40876I MAINT: NO RECOVER PLUS PTFs APPLIED
BMC96010I SOLUTION COMMON CODE V10.1.00
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFs APPLIED
BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276
BMC40876I MAINT: BPJ0279 BPJ0288
BMC40001I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
```

Figure 57 Example 2A—AFRSUMRY for a simple REBUILD INDEX

```
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40011I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40601I SOLUTION COMMON CODE V10.1.00
BMC40876I UTILITY EXECUTION COMPLETE 01/13/2011 08:35:56, RETURN CODE = 4
BMC40876I MAINT: NO RECOVER PLUS PTFs APPLIED
BMC96010I SOLUTION COMMON CODE V10.1.00
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFs APPLIED
BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276
BMC40876I MAINT: BPJ0279 BPJ0288
BMC40001I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
BMC40001I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40601I SOLUTION COMMON CODE V10.1.00
BMC40876I UTILITY EXECUTION COMPLETE 01/13/2011 08:35:56, RETURN CODE = 4
BMC40876I MAINT: NO RECOVER PLUS PTFs APPLIED
BMC96010I SOLUTION COMMON CODE V10.1.00
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFs APPLIED
BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276
BMC40876I MAINT: BPJ0279 BPJ0288
BMC40001I UTILITY EXECUTION STARTING 01/13/2011 08:35:36
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
```

Figure 58 Example 2A—AFRSTMT for a simple REBUILD INDEX (part 1 of 3)
### Example 2A—AFRSTMT for a simple REBUILD INDEX (part 2 of 3)

<table>
<thead>
<tr>
<th>BMC40927I</th>
<th>RDB2STAT = YES</th>
<th>(RESET DB2 OBJECT STATUS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40946I</td>
<td>AMSCAT = NO</td>
<td>(USE 'CATALOG' IN IDCAMS INPUT)</td>
</tr>
<tr>
<td>BMC40928I</td>
<td>SMCORE = ( OK, OK)</td>
<td>(SORT CORE VALUES)</td>
</tr>
<tr>
<td>BMC40893I</td>
<td>RESINV = ( OK)</td>
<td>(MEMORY BELOW 1MB EXCLUDED FROM USE BY SORT)</td>
</tr>
<tr>
<td>BMC40938I</td>
<td>WKUNIT = SYSALDDA</td>
<td>(WORK UNIT NAME)</td>
</tr>
<tr>
<td>BMC40956I</td>
<td>RCLTSK = 10</td>
<td>(MAXIMUM RECALL TASKS)</td>
</tr>
<tr>
<td>BMC40411I</td>
<td>IXRECP = NO</td>
<td>(SET INDEX RECOVER PENDING)</td>
</tr>
<tr>
<td>BMC40891I</td>
<td>MAXDRIVES = 0</td>
<td>(DEFAULT MAX TAPE DRIVES)</td>
</tr>
<tr>
<td>BMC40942I</td>
<td>MAXSORT = 6</td>
<td>(DEFAULT MAX CONCURRENT INDEX SORTS)</td>
</tr>
<tr>
<td>BMC96230I</td>
<td>KSORTSHARE = YES</td>
<td>(KEY SORTS SHARED BY ALL TASKS)</td>
</tr>
<tr>
<td>BMC96192I</td>
<td>MAXLSORT = 50</td>
<td>(MAX CONCURRENT LOG SORTS)</td>
</tr>
<tr>
<td>BMC40962I</td>
<td>SORTDEV = SORT DEFAULT</td>
<td>(DEVICE TYPE FOR LOG AND INDEX SORTS)</td>
</tr>
<tr>
<td>BMC40963I</td>
<td>SORTNUM = SORT DEFAULT</td>
<td>(WORK DATASETS FOR LOG AND INDEX SORTS)</td>
</tr>
<tr>
<td>BMC40964I</td>
<td>OUTCOPY = ASCODED</td>
<td>(OUTCOPY CREATION DIRECTIVE)</td>
</tr>
<tr>
<td>BMC40547I</td>
<td>INDEXLOG = NO</td>
<td>(DEFAULT USE OF INDEXLOG STRATEGY)</td>
</tr>
<tr>
<td>BMC40537I</td>
<td>ERRCONT = 10</td>
<td>(DEFAULT MAX SEVERE ERRORS)</td>
</tr>
<tr>
<td>BMC40241I</td>
<td>PUBLICPLAN = YES</td>
<td>(GRANT PLAN TO PUBLIC)</td>
</tr>
<tr>
<td>BMC96221I</td>
<td>AUTOSIZE = YES</td>
<td>(AUTO SIZE ACCUM OUTPUT FILES)</td>
</tr>
<tr>
<td>BMC40241I</td>
<td>PUBLICPLAN = YES</td>
<td>(GRANT PLAN TO PUBLIC)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE01 = 3590-1</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE02 = 3490</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE03 = 3480X</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE04 = ATLYTS</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE05 = ATLYTS</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE06 = CARTVTS</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE07 = CARTVTS</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE08 = CRT9NS</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE09 = CRT9NS</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE10 = DANBARR</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE11 = EXT3592</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE12 = IBMATL</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE13 = IBMATL</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE14 = SA3590</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE15 = TAPEABL</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE16 = 3590E</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE17 = 3590E</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE18 = 3480</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE19 = 3400-9</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE20 = SYS3480R</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40967I</td>
<td>TAPE DEVICE21 = SYS3480R</td>
<td>(TAPE DEVICE)</td>
</tr>
<tr>
<td>BMC40929I</td>
<td>SORT VALUES: SORTCORE = 0 K</td>
<td>(SORT MEMORY VALUE)</td>
</tr>
<tr>
<td>BMC40947I</td>
<td>VSCORET = 0 K</td>
<td>(TOTAL MEMORY OVERRIDE)</td>
</tr>
<tr>
<td>BMC96111I</td>
<td>DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE</td>
<td></td>
</tr>
<tr>
<td>BMC9613I</td>
<td>COPIES = (LP, LB)</td>
<td>(DEFAULT SEQUENCE FOR IMAGE COPIES)</td>
</tr>
<tr>
<td>BMC9614I</td>
<td>ACCUMS = (LP, LB)</td>
<td>(DEFAULT SEQUENCE FOR CA FILES)</td>
</tr>
<tr>
<td>BMC9612I</td>
<td>DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE</td>
<td></td>
</tr>
<tr>
<td>BMC9611I</td>
<td>COPIES = (RP, RB)</td>
<td>(DEFAULT SEQUENCE FOR IMAGE COPIES)</td>
</tr>
<tr>
<td>BMC9611I</td>
<td>ACCUMS = (RP, RB)</td>
<td>(DEFAULT SEQUENCE FOR CA FILES)</td>
</tr>
<tr>
<td>BMC40018I</td>
<td>RECOVER PLUS FOR DB2 V10.1.00</td>
<td></td>
</tr>
<tr>
<td>BMC40876I</td>
<td>MAINT: NO RECOVER PLUS PTFs APPLIED</td>
<td></td>
</tr>
<tr>
<td>BMC9610I</td>
<td>SOLUTION COMMON CODE V10.1.00</td>
<td></td>
</tr>
<tr>
<td>BMC40876I</td>
<td>MAINT: NO SOLUTION COMMON CODE PTFs APPLIED</td>
<td></td>
</tr>
<tr>
<td>BMC9610I</td>
<td>BMCSORT ENGINE V2.3.01</td>
<td></td>
</tr>
<tr>
<td>BMC40876I</td>
<td>MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276</td>
<td></td>
</tr>
<tr>
<td>BMC40876I</td>
<td>MAINT: BPJ0279 BPJ0288</td>
<td></td>
</tr>
<tr>
<td>BMC40611I</td>
<td>INPUT STATEMENTS:</td>
<td></td>
</tr>
<tr>
<td>BMC4010I</td>
<td>OPTION EARLYRECALL INDEXLOG NO</td>
<td></td>
</tr>
<tr>
<td>BMC4010I</td>
<td>RECOVER TABLESPACE AFRX01.TSEX1P1</td>
<td></td>
</tr>
<tr>
<td>BMC4010I</td>
<td>REBUILD INDEX (AFR.INEX1P1 , AFR.ICEX1P1) NOWORKDDN</td>
<td></td>
</tr>
</tbody>
</table>
Example 2A: Using REBUILD INDEX

BMC40960I  INDEXLOG STRATEGY DISABLED
BMC40351I  ANALYZE STARTING 01/13/2011 08:35:37
BMC40351I  CATALOG LOOKUP STARTING 01/13/2011 08:35:37
BMC40631I  USING STANDARD DB2 SECURITY
BMC40705I  VOLUMES IN STOGROUP SYSDFTL WILL BE USED IN THE FOLLOWING ORDER:
BMC40707I  (<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>)
BMC40361I  CATALOG LOOKUP FINISHED.
BMC40287I  ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I  CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40335I  SYNCHRONIZATION STARTING 01/13/2011 08:35:37
BMC40336I  SYNCHRONIZATION FINISHED.
BMC40287I  ELAPSED TIME = 00:00:02, UTILITY ELAPSED TIME = 00:00:03
BMC40288I  CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC96306I  MAXLSORT REDUCED FROM 50 TO 18 DUE TO MEMORY CONSTRAINTS

BMC40351I  PLANNING STARTING 01/13/2011 08:35:40
BMC40351I  PLAN EXECUTION STARTING 01/13/2011 08:35:43

BMC40483I  BUILD COMPLETE.
BMC40301I  TABLESPACE = AFREX01.TSEX1P1
BMC40269I  PARTITION = 1
BMC40483I  BUILD COMPLETE.
BMC40301I  TABLESPACE = AFREX01.TSEX1P1
BMC40269I  PARTITION = 2
BMC40483I  BUILD COMPLETE.
BMC40303I  INDEX = AFR.ICEX1P1
BMC40269I  PARTITION = 1
BMC40483I  MERGE COMPLETE.
BMC40301I  TABLESPACE = AFREX01.TSEX1P1
BMC40269I  PARTITION = 3
BMC40483I  BUILD COMPLETE.
BMC40303I  INDEX = AFR.ICEX1P1
BMC40269I  PARTITION = 2
BMC40483I  MERGE COMPLETE.
BMC40301I  TABLESPACE = AFREX01.TSEX1P1
BMC40269I  PARTITION = 4
BMC40483I  BUILD COMPLETE.
BMC40303I  INDEX = AFR.ICEX1P1
BMC40269I  PARTITION = 3
BMC40483I  MERGE COMPLETE.
BMC40301I  TABLESPACE = AFREX01.TSEX1P1
BMC40269I  PARTITION = 5
BMC40483I  BUILD COMPLETE.
BMC40303I  INDEX = AFR.ICEX1P1
BMC40269I  PARTITION = 4
BMC40483I  MERGE COMPLETE.
BMC40301I  TABLESPACE = AFREX01.TSEX1P1
BMC40269I  PARTITION = 6
BMC40483I  MERGE COMPLETE.
Example 2A: Using REBUILD INDEX

Figure 59  Example 2A—AFRPRINT for a simple REBUILD INDEX  (part 2 of 4)
Figure 59  Example 2A—AFRPRINT for a simple REBUILD INDEX (part 3 of 4)

<table>
<thead>
<tr>
<th>BMC402691</th>
<th>PARTITION = 13</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC404831</td>
<td>MERGE COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 16</td>
</tr>
<tr>
<td>BMC404831</td>
<td>MERGE COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 17</td>
</tr>
<tr>
<td>BMC404831</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 15</td>
</tr>
<tr>
<td>BMC404831</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC403031</td>
<td>INDEX = AFR.ICEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 16</td>
</tr>
<tr>
<td>BMC404831</td>
<td>MERGE COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 18</td>
</tr>
<tr>
<td>BMC404831</td>
<td>MERGE COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 19</td>
</tr>
<tr>
<td>BMC404831</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC403031</td>
<td>INDEX = AFR.ICEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 15</td>
</tr>
<tr>
<td>BMC404831</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC403031</td>
<td>INDEX = AFR.ICEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 18</td>
</tr>
<tr>
<td>BMC404831</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 17</td>
</tr>
<tr>
<td>BMC404831</td>
<td>MERGE COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 20</td>
</tr>
<tr>
<td>BMC404831</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC403031</td>
<td>INDEX = AFR.ICEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 19</td>
</tr>
<tr>
<td>BMC404831</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC403031</td>
<td>INDEX = AFR.ICEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 20</td>
</tr>
<tr>
<td>BMC404831</td>
<td>MERGE COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 21</td>
</tr>
<tr>
<td>BMC404831</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 19</td>
</tr>
<tr>
<td>BMC404831</td>
<td>MERGE COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 22</td>
</tr>
<tr>
<td>BMC404831</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC403031</td>
<td>INDEX = AFR.ICEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 22</td>
</tr>
<tr>
<td>BMC404831</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC403031</td>
<td>INDEX = AFR.ICEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 19</td>
</tr>
<tr>
<td>BMC404831</td>
<td>MERGE COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 23</td>
</tr>
<tr>
<td>BMC404831</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 21</td>
</tr>
<tr>
<td>BMC404831</td>
<td>MERGE COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 24</td>
</tr>
<tr>
<td>BMC404831</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC403031</td>
<td>INDEX = AFR.ICEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 21</td>
</tr>
<tr>
<td>BMC404831</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC402691</td>
<td>PARTITION = 23</td>
</tr>
</tbody>
</table>
Example 2B: Using RECOVER INDEX

In the second job in this example (Figure 60), the index is recovered from image copies and log. Note that this job uses MSGLEVEL(2), which produces all of the output produced by the default of MSGLEVEL(1) and the AFRPLAN.

Figure 61 through Figure 65 on page 353 show the following output files produced by this JCL:

- AFRSUMRY
- AFRSTMT
- AFRPRINT
- AFROSUM
- AFRPLAN

See Table 10 on page 277 for a description of each of these files.
Example 2B: Using RECOVER INDEX

Figure 61  Example 2B—AFRSUMRY for a simple RECOVER INDEX (part 2 of 2)

<table>
<thead>
<tr>
<th>BMC40018I</th>
<th>RECOVER PLUS FOR DB2 V10.1.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40876I</td>
<td>MAINT: NO RECOVER PLUS PTFS APPLIED</td>
</tr>
<tr>
<td>BMC96010I</td>
<td>SOLUTION COMMON CODE V10.1.00</td>
</tr>
<tr>
<td>BMC40876I</td>
<td>MAINT: NO SOLUTION COMMON CODE PTFS APPLIED</td>
</tr>
<tr>
<td>BMC06010I</td>
<td>BMCSORT ENGINE V2.3.01</td>
</tr>
<tr>
<td>BMC40876I</td>
<td>MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276</td>
</tr>
<tr>
<td>BMC40876I</td>
<td>MAINT: BPJ0279 BPJ0288</td>
</tr>
<tr>
<td>BMC40002I</td>
<td>UTILITY ID = MVSMAR3.AFR02B. DB2 SUBSYSTEM ID = DXW</td>
</tr>
<tr>
<td>BMC40094I</td>
<td>ATTEMPTING TO CONNECT TO DB2 SUBSYSTEM DXW. USING PLAN AFRB101T</td>
</tr>
<tr>
<td>BMC40342I</td>
<td>A GROUP ATTACH WAS PERFORMED WITH DXW</td>
</tr>
<tr>
<td>BMC40067I</td>
<td>DB2 W4 IS MEMBER DXW4 IN DATA SHARING GROUP DSNDXW</td>
</tr>
<tr>
<td>BMC40475I</td>
<td>INPUT STATEMENTS PRINTED IN AFRSTMT</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>ANALYZE FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:03, UTILITY ELAPSED TIME = 00:00:04</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>PLANNING FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:03, UTILITY ELAPSED TIME = 00:00:08</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>PLAN EXECUTION FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:18, UTILITY ELAPSED TIME = 00:00:26</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:01</td>
</tr>
<tr>
<td>BMC40476I</td>
<td>EXECUTION SUMMARY</td>
</tr>
<tr>
<td>BMC40477I</td>
<td>TABLE SPACES / PHASES (PARTITIONS) RECOVERED: 1 (24)</td>
</tr>
<tr>
<td>BMC40478I</td>
<td>INDEX SPACES / PHASES (PARTITIONS) RECOVERED: 2 (24)</td>
</tr>
<tr>
<td>BMC40005I</td>
<td>UTILITY EXECUTION COMPLETE 01/03/2011 18:32:24, RETURN CODE = 0</td>
</tr>
</tbody>
</table>

Figure 62  Example 2B—AFRSTMT for a simple RECOVER INDEX (part 1 of 4)

<table>
<thead>
<tr>
<th>BMC40018I</th>
<th>RECOVER PLUS FOR DB2 V10.1.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40473I</td>
<td>COPYRIGHT BMC SOFTWARE INC. 1991-2011</td>
</tr>
<tr>
<td>BMC96173I</td>
<td>RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884</td>
</tr>
<tr>
<td>BMC40474I</td>
<td>CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO <a href="mailto:SUPPORT@BMC.COM">SUPPORT@BMC.COM</a></td>
</tr>
<tr>
<td>BMC40937I</td>
<td>z/OS RELEASE = 1.9.00, PID = H887740, SMS RELEASE = 1.9.0</td>
</tr>
<tr>
<td>BMC40157I</td>
<td>AVAILABLE REGION BELOW 16M = 8836K, AVAILABLE REGION ABOVE 16M = 1574392K, NUMBER OF CPUS = 3</td>
</tr>
<tr>
<td>BMC40020I</td>
<td>BYT:0, 0, 0; BLK:0, 0, 0; TOT:547388, 547388, 547388; iter: 1; RC:4</td>
</tr>
<tr>
<td>BMC40093I</td>
<td>PARM LIST: DB2 SSID = DXW</td>
</tr>
<tr>
<td>BMC40930I</td>
<td>PARM NOT CODED - UTILID</td>
</tr>
<tr>
<td>BMC40099I</td>
<td>UTILITY NEW/RESTART REQUESTED - SYNCPOINT RESTART WILL OCCUR IF UTILID IS IN BMCUTIL</td>
</tr>
<tr>
<td>BMC40933I</td>
<td>RESTART = NEW/RESTART</td>
</tr>
<tr>
<td>BMC40934I</td>
<td>MSGLEVEL = 2</td>
</tr>
<tr>
<td>BMC40930I</td>
<td>PARM NOT CODED - CKPT</td>
</tr>
<tr>
<td>BMC40930I</td>
<td>PARM NOT CODED - RDB2STAT</td>
</tr>
<tr>
<td>BMC40930I</td>
<td>PARM NOT CODED - AFROPTS</td>
</tr>
<tr>
<td>BMC40930I</td>
<td>PARM NOT CODED - ACAOPTS</td>
</tr>
<tr>
<td>BMC40924I</td>
<td>OPTIONS: PLANRECV = AFRB092T (MAIN PLAN)</td>
</tr>
<tr>
<td>BMC40925I</td>
<td>TBUFFS = 1000 (BUFFER MANAGER PAGES)</td>
</tr>
<tr>
<td>BMC40944I</td>
<td>OPNDB2ID = YES (ACQUIRE RACF AUTHORITY OF DB2)</td>
</tr>
<tr>
<td>BMC40926I</td>
<td>CHECKPT = PHASE (CHECKPOINTING)</td>
</tr>
<tr>
<td>BMC40945I</td>
<td>CHECKINT = 0 (CHECKPOINT INTERVAL MINUTES)</td>
</tr>
<tr>
<td>BMC40927I</td>
<td>RDB2STAT = YES (RESET DB2 OBJECT STATUS)</td>
</tr>
<tr>
<td>BMC40946I</td>
<td>AMSCAT = NO (USE 'CATALOG' IN IDCAMS INPUT)</td>
</tr>
<tr>
<td>BMC40928I</td>
<td>SMCORE = (OK, OK) (SORT CORE VALUES)</td>
</tr>
<tr>
<td>BMC40893I</td>
<td>RESINV = (OK) (MEMORY BELOW 16MB EXCLUDED FROM USE BY SDRT)</td>
</tr>
<tr>
<td>BMC40938I</td>
<td>WKUNIT = SYSALD (WORK UNIT NAME)</td>
</tr>
</tbody>
</table>
### Example 2B—AFRSTMT for a simple RECOVER INDEX (part 2 of 4)

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCLTSK = 10</td>
<td>Maximum recall tasks</td>
</tr>
<tr>
<td>IXREEP = NO</td>
<td>Set index recovery pending</td>
</tr>
<tr>
<td>MAXDRIVES = 0</td>
<td>Default maximum tape drives</td>
</tr>
<tr>
<td>MAXLOGS = 3</td>
<td>Default maximum concurrent log files</td>
</tr>
<tr>
<td>MAXSORT = 6</td>
<td>Default maximum concurrent index sorts</td>
</tr>
<tr>
<td>KSORTSHARE = YES</td>
<td>Key sorts shared by all tasks</td>
</tr>
<tr>
<td>MAXSORT = 50</td>
<td>Maximum concurrent log sorts</td>
</tr>
<tr>
<td>SORTDEV = SORT DEFAULT</td>
<td>Device type for log and index sorts</td>
</tr>
<tr>
<td>SORTNUM = SORT DEFAULT</td>
<td>Work datasets for log and index sorts</td>
</tr>
<tr>
<td>OUTCOPY = ASCODED</td>
<td>Outcopy creation directive</td>
</tr>
<tr>
<td>INDEXLOG = NO</td>
<td>Default use of indexlog strategy</td>
</tr>
<tr>
<td>WTOR = YES</td>
<td>Issue WTOR if space remains in stopp status</td>
</tr>
<tr>
<td>ERRCONT = 10</td>
<td>Default maximum severe errors</td>
</tr>
<tr>
<td>AUTO SIZE = YES</td>
<td>Auto size accum output files</td>
</tr>
<tr>
<td>SORTDEVT = SORT DEFAULT</td>
<td>Device type for log and index sorts</td>
</tr>
<tr>
<td>TAPE DEVICE = 3590-1</td>
<td>Tape device 1</td>
</tr>
<tr>
<td>TAPE DEVICE = 3490</td>
<td>Tape device 2</td>
</tr>
<tr>
<td>TAPE DEVICE = 3480X</td>
<td>Tape device 3</td>
</tr>
<tr>
<td>TAPE DEVICE = ATLVTS</td>
<td>Tape device 4</td>
</tr>
<tr>
<td>TAPE DEVICE = ATL3592</td>
<td>Tape device 5</td>
</tr>
<tr>
<td>TAPE DEVICE = CART</td>
<td>Tape device 6</td>
</tr>
<tr>
<td>TAPE DEVICE = CARTVTS</td>
<td>Tape device 7</td>
</tr>
<tr>
<td>TAPE DEVICE = CRTLWS</td>
<td>Tape device 8</td>
</tr>
<tr>
<td>TAPE DEVICE = CRTVTS</td>
<td>Tape device 9</td>
</tr>
<tr>
<td>TAPE DEVICE = DABARR</td>
<td>Tape device 10</td>
</tr>
<tr>
<td>TAPE DEVICE = EXT3592</td>
<td>Tape device 11</td>
</tr>
<tr>
<td>TAPE DEVICE = IBMATL</td>
<td>Tape device 12</td>
</tr>
<tr>
<td>TAPE DEVICE = IBMIVS</td>
<td>Tape device 13</td>
</tr>
<tr>
<td>TAPE DEVICE = SA3590</td>
<td>Tape device 14</td>
</tr>
<tr>
<td>TAPE DEVICE = TAPEABL</td>
<td>Tape device 15</td>
</tr>
<tr>
<td>TAPE DEVICE = 3590B</td>
<td>Tape device 16</td>
</tr>
<tr>
<td>TAPE DEVICE = 3480</td>
<td>Tape device 17</td>
</tr>
<tr>
<td>TAPE DEVICE = 3400-9</td>
<td>Tape device 18</td>
</tr>
<tr>
<td>TAPE DEVICE = SYS3480R</td>
<td>Tape device 20</td>
</tr>
<tr>
<td>TAPE DEVICE = SYS348XR</td>
<td>Tape device 21</td>
</tr>
<tr>
<td>SORTCORE = 0 K</td>
<td>Sort core value</td>
</tr>
<tr>
<td>VSCORET = 0 K</td>
<td>Total memory override</td>
</tr>
<tr>
<td>DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE</td>
<td>Default sequence for local site</td>
</tr>
<tr>
<td>DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE</td>
<td>Default sequence for remote site</td>
</tr>
<tr>
<td>RECOVER PLUS FOR DB2 V10.1.00</td>
<td>Recover plus for DB2</td>
</tr>
<tr>
<td>MAINT: NO RECOVER PLUS PTFS APPLIED</td>
<td>Maintenance not applying recover plus ptfs</td>
</tr>
<tr>
<td>SOLUTION COMMON CODE V10.1.00</td>
<td>Solution common code</td>
</tr>
<tr>
<td>MAINT: NO SOLUTION COMMON CODE PTFS APPLIED</td>
<td>Maintenance not applying solution common code ptfs</td>
</tr>
<tr>
<td>PACKAGE CHECKING AND BINDING STARTING 01/03/2011 18:31:57</td>
<td>Package checking and binding started</td>
</tr>
<tr>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00</td>
<td>Elapsed time</td>
</tr>
<tr>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
<td>Utility CPU time</td>
</tr>
</tbody>
</table>
Figure 62  Example 2B—AFRSTMT for a simple RECOVER INDEX  (part 3 of 4)

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40101I</td>
<td>OPTION EARLYRECALL INDEXLOG YES</td>
</tr>
<tr>
<td>BMC40101I</td>
<td>RECOVER TABLESPACE AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40101I</td>
<td>RECOVER INDEX (AFR.INEX1P1 , AFR.ICEX1P1)</td>
</tr>
<tr>
<td>BMC40101I</td>
<td>INDEXLOG YES STRATEGY REQUESTED</td>
</tr>
<tr>
<td>BMC40335I</td>
<td>ANALYZE STARTING 01/03/2011 18:31:58</td>
</tr>
<tr>
<td>BMC40335I</td>
<td>CATALOG LOOKUP STARTING 01/03/2011 18:31:58</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>PROC RO STM STARTED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>SCAN SYSTB FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>SCAN SYSIX FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>SCAN SYSKY FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>COL LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>SCAN SYSFL FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>TS LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>DB LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>TB LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>IP LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>SG LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40705I</td>
<td>VOLUMES IN STOGROUP SYSDFLT WILL BE USED IN THE FOLLOWING ORDER:</td>
</tr>
<tr>
<td>BMC40707I</td>
<td>(&lt;SMS VOLUME&gt;,&lt;SMS VOLUME&gt;,&lt;SMS VOLUME&gt;,&lt;SMS VOLUME&gt;,&lt;SMS VOLUME&gt;,&lt;SMS VOLUME&gt;)</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>SCAN SYSVOL FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>CHK DS FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40336I</td>
<td>HDR PG RD FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01</td>
</tr>
</tbody>
</table>
Example 2B—AFRPRINT for a simple RECOVER INDEX (part 4 of 4)

<table>
<thead>
<tr>
<th>BMC40288I</th>
<th>CPU TIME     = 00:00:00, UTILITY CPU TIME = 00:00:00</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40336I</td>
<td>SYSCOPY LOOKUP FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME     = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
</tbody>
</table>

| BMC40336I | CATALOG LOOKUP FINISHED. |
| BMC40287I | ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01 |
| BMC40288I | CPU TIME     = 00:00:00, UTILITY CPU TIME = 00:00:00 |

| BMC40335I | SYNCHRONIZATION STARTING 01/03/2011 18:31:59 |
| BMC40025I | OUTPUT FROM COMMAND '-STO DB(AFREX01) SPACE(TSEX1P1) AT (COMMIT)' FOLLOWS: |
| BMC40020I | DSN9022I  *DXW4 DSNTDDIS 'STOP DATABASE' NORMAL COMPLETION |
| BMC40021I | DSN9022I  *DXW4 DSNTDDIS 'START DATABASE' NORMAL COMPLETION |

| BMC40336I | SYNCHRONIZATION FINISHED. |
| BMC40287I | ELAPSED TIME = 00:00:02, UTILITY ELAPSED TIME = 00:00:04 |
| BMC40288I | CPU TIME     = 00:00:00, UTILITY CPU TIME = 00:00:00 |

| BMC40066I | TIME TO READ SYSLGRNX WAS 00:00:00 |

| BMC40336I | DDSCAN FINISHED. |
| BMC40287I | ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:04 |
| BMC40288I | CPU TIME     = 00:00:00, UTILITY CPU TIME = 00:00:00 |

| BMC40335I | PLANNING STARTING 01/03/2011 18:32:02 |
| BMC40459I | BSDS DSNDXW.DXW4.BSDS01 SELECTED |
| BMC40061I | TIME TO READ SYSLGRNX WAS 00:00:00 |

| BMC40335I | PLAN EXECUTION STARTING 01/03/2011 18:32:05 |
| BMC40025I | OUTPUT FROM COMMAND '-STA DB(AFREX01) SPACE(TSEX1P1)  ACCESS(RW)' FOLLOWS: |
| BMC40020I | DSN9022I  *DXW4 DSNTDDIS 'START DATABASE' NORMAL COMPLETION |
| BMC40021I | DSN9022I  *DXW4 DSNTDDIS 'START DATABASE' NORMAL COMPLETION |

| BMC40335I | PLAN EXECUTION FINISHED |
| BMC40102I | MERGE PHASE STARTING 01/03/2011 18:32:05 |
| BMC40593I | INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00 |

| BMC40018I | RECOVER PLUS FOR DB2 V10.1.00 |
| BMC40795I | FULL COPY DATASET IS MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00 |
| BMC40593I | INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00 |

| BMC40016I | MERGE PHASE STARTING 01/03/2011 18:32:05 |
| BMC40799I | MERGE STARTING FOR DATA SET DSNDXW.DSNDBD.AFREX01.TSEX1P1.I0001.A001 |

| BMC40102I | MERGE TABLESPACE STEP STARTING |
| BMC40016I | DATA SET MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00083 ELAPSED = 00:00:00 |
| BMC40016I | PENDING FLAG RECP SET FOR |
| BMC40593I | INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00 |

| BMC40016I | DATA SET MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00083 ELAPSED = 00:00:00 |
| BMC40016I | PENDING FLAG RECP SET FOR |
| BMC40305I | DSNUM = 1 |
| BMC40345I | SPACE = AFREX01.TSEX1P1 |

| BMC40305I | DSNUM = 1 |
| BMC40350I | AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBD.AFREX01.TSEX1P1.I0001.A001 |
| BMC40356I | AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.TSEX1P1.I0001.A001 |
| BMC40356I | DATA SET DSNDXW.DSNDBD.AFREX01.TSEX1P1.I0001.A001 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00087 ELAPSED = 00:00:00 |

| BMC40345I | SPACE = AFREX01.TSEX1P1 |
| BMC40593I | INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00 |

| BMC40301I | TABLESPACE = AFREX01.TSEX1P1 |
| BMC40305I | DSNUM = 1 |
| BMC40350I | DATA SET DSNDXW.DSNDBD.AFREX01.TSEX1P1.I0001.A001 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00087 ELAPSED = 00:00:00 |

| BMC40102I | INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00 |
| BMC40016I | DATA SET MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00083 ELAPSED = 00:00:00 |

| BMC40016I | PENDING FLAG RECP SET FOR |
| BMC40305I | DSNUM = 1 |
| BMC40350I | DATA SET DSNDXW.DSNDBD.AFREX01.TSEX1P1.I0001.A001 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00087 ELAPSED = 00:00:00 |

| BMC40593I | INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00 |

Figure 62 Example 2B—AFRPRINT for a simple RECOVER INDEX (part 4 of 6)
Figure 63  Example 2B—Edited AFRPRINT for a simple RECOVER INDEX (part 2 of 6)

<table>
<thead>
<tr>
<th>BMC40016I</th>
<th>MERGE PHASE STARTING 01/03/2011 18:32:06</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40030I</td>
<td>INDEX = AFR.INEX1P1</td>
</tr>
<tr>
<td>BMC40030I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40350I</td>
<td>DATA SET MVSMAR3.LP.AFREX01.INEX1P1.F001.G0008V00 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00999</td>
</tr>
<tr>
<td>BMC40060I</td>
<td>PENDING FLAG RSEP SET FOR</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.INEX1P1</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.INEX1P1.I0001.A001</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.INEX1P1.I0001.A001</td>
</tr>
<tr>
<td>BMC40350I</td>
<td>DATA SET DSNDXW.DSNDBC.AFREX01.INEX1P1.I0001.A001 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00999</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFRICLX1P1</td>
</tr>
<tr>
<td>BMC40016I</td>
<td>MERGE PHASE STARTING 01/03/2011 18:32:07</td>
</tr>
<tr>
<td>BMC40030I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40350I</td>
<td>DATA SET MVSMAR3.LP.AFREX01.INEX1P1.F001.G0008V00 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00999</td>
</tr>
<tr>
<td>BMC40060I</td>
<td>PENDING FLAG RSEP SET FOR</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.INEX1P1</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.INEX1P1.I0001.A001</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.INEX1P1.I0001.A001</td>
</tr>
<tr>
<td>BMC40350I</td>
<td>DATA SET DSNDXW.DSNDBC.AFREX01.INEX1P1.I0001.A001 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00999</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFRICLX1P1</td>
</tr>
<tr>
<td>BMC40016I</td>
<td>MERGE PHASE STARTING 01/03/2011 18:32:07</td>
</tr>
<tr>
<td>BMC40030I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40350I</td>
<td>DATA SET MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0000V00 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00999</td>
</tr>
<tr>
<td>BMC40060I</td>
<td>PENDING FLAG RSEP SET FOR</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.TSEX1P1.I0001.A001</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.TSEX1P1.I0001.A001</td>
</tr>
<tr>
<td>BMC40350I</td>
<td>DATA SET DSNDXW.DSNDBC.AFREX01.TSEX1P1.I0001.A001 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00999</td>
</tr>
</tbody>
</table>
Figure 63  Example 2B—Edited AFRPRINT for a simple RECOVER INDEX  (part 3 of 6)

| BMC40593I | START MERGE COMPLETE, ELAPSED TIME = 00:00:00 |
| BMC40102I | MERGE INDEXSPACE STEP STARTING |
| BMC40795I | FULL COPY DATASET IS MVSMAR3.LP.AFREX01.ICEX1P1.F001.G0008V00 |
| BMC40799I | MERGE STARTING FOR DATA SET DSNDXW.DSNDBD.AFREX01.ICEX1P1.I0001.A001 |
| BMC40593I | INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00 |
| BMC40593I | SPIN COMPLETE, ELAPSED TIME = 00:00:00 |
| BMC40902I | TOTAL WAITS = 2, TOTAL WAIT TIME = 00:00:00 |
| BMC40981I | FULL IC MVSMAR3.LP.AFREX01.ICEX1P1.F001.G0008V00 READ WAITS = 1, WAIT TIME = 00:00:00 |
| BMC40905I | SPACE WRITE WAITS = 1, WAIT TIME = 00:00:00 |
| BMC40985I | NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 11, USED = 11 |
| BMC40790I | NUMBER OF PAGES READ FROM FULL COPY TOTAL = 11, USED = 11 |
| BMC40798I | NUMBER OF PAGES OUTPUT IS = 17 |
| BMC40900I | MERGE INDEXSPACE STEP COMPLETE |
| BMC40901I | DDNAME SYS00097 (DSN DSNDXW.DSNDBD.AFREX01.ICEX1P1.I0001.A001) WAS SUCCESSFULLY UNALLOCATED |
| BMC40350I | DATA SET MVSMAR3.LP.AFREX01.ICEX1P1.F002.G0008V00 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00099 ELAPSED = 00:00:00 |
| BMC40601I | PENDING FLAG RBDP RESET FOR |
| BMC40345I | SPACE = AFREX01.ICEX1P1 |
| BMC40305I | DSNM = 2 |
| BMC40091I | 01/03/2011 18:32:07 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE |
| BMC40012I | MERGE PHASE COMPLETE. ELAPSED TIME = 00:00:00, TIME SINCE UTILITY START = 00:00:10 |
| BMC40868I | MERGE PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00 |

| BMC40016I | MERGE PHASE STARTING 01/03/2011 18:32:07 |
| BMC40303I | INDEX = AFREX01.ICEX1P1 |
| BMC40305I | DSNM = 2 |
| BMC40350I | DATA SET MVSMAR3.LP.AFREX01.ICEX1P1.F002.G0008V00 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00099 ELAPSED = 00:00:00 |
| BMC40601I | PENDING FLAG RBDP SET FOR |
| BMC40345I | SPACE = AFREX01.ICEX1P1 |
| BMC40305I | DSNM = 2 |
| BMC40350I | AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBD.AFREX01.ICEX1P1.I0001.A002 |
| BMC40350I | AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBD.AFREX01.ICEX1P1.I0001.A002 |
| BMC40350I | DATA SET DSNDXW.DSNDBD.AFREX01.ICEX1P1.I0001.A002 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00103 ELAPSED = 00:00:00 |
| BMC96304I | THIS PHASE WILL RUN IN EXECUTION SUBTASK 1 |
| BMC40593I | START MERGE COMPLETE, ELAPSED TIME = 00:00:00 |
| BMC40102I | MERGE INDEXSPACE STEP STARTING |
| BMC40795I | FULL COPY DATASET IS MVSMAR3.LP.AFREX01.ICEX1P1.F001.G0008V00 |
| BMC40799I | MERGE STARTING FOR DATA SET DSNDXW.DSNDBD.AFREX01.ICEX1P1.I0001.A002 |
| BMC40593I | INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00 |
| BMC40593I | SPIN COMPLETE, ELAPSED TIME = 00:00:00 |
| BMC40902I | TOTAL WAITS = 2, TOTAL WAIT TIME = 00:00:00 |
| BMC40981I | FULL IC MVSMAR3.LP.AFREX01.ICEX1P1.F002.G0008V00 READ WAITS = 1, WAIT TIME = 00:00:00 |
| BMC40905I | SPACE WRITE WAITS = 1, WAIT TIME = 00:00:00 |
| BMC40985I | NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 11, USED = 11 |
| BMC40790I | NUMBER OF PAGES READ FROM FULL COPY TOTAL = 11, USED = 11 |
| BMC40798I | NUMBER OF PAGES OUTPUT IS = 17 |
| BMC40900I | MERGE INDEXSPACE STEP COMPLETE |
| BMC40901I | DDNAME SYS00099 (DSN MVSMAR3.LP.AFREX01.ICEX1P1.F002.G0008V00) WAS SUCCESSFULLY UNALLOCATED |
| BMC40061I | PENDING FLAG RBDP RESET FOR |
| BMC40345I | SPACE = AFREX01.ICEX1P1 |
| BMC40305I | DSNM = 2 |
| BMC40091I | 01/03/2011 18:32:09 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE |
| BMC40012I | MERGE PHASE COMPLETE. ELAPSED TIME = 00:00:01, TIME SINCE UTILITY START = 00:00:11 |
| BMC40868I | MERGE PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00 |

| BMC40016I | MERGE PHASE STARTING 01/03/2011 18:32:08 |
| BMC40303I | INDEX = AFREX01.ICEX1P1 |
| BMC40305I | DSNM = 3 |
| BMC40350I | DATA SET MVSMAR3.LP.AFREX01.ICEX1P1.F003.G0008V00 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00104 ELAPSED = 00:00:00 |
| BMC40601I | PENDING FLAG RBDP SET FOR |
| BMC40345I | SPACE = AFREX01.ICEX1P1 |
BMC40305I DSNUM = 3
BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.ICEXIP1.10001.A003
BMC40356I AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.ICEXIP1.10001.A003
BMC40350I DATA SET DSNDXW.DSNDBD.AFREX01.ICEXIP1.10001.A003 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00108 ELAPSED = 00:00:00
BMC6304I THIS PHASE WILL RUN IN EXECUTION SUBTASK 1
BMC40593I START MERGE COMPLETE, ELAPSED TIME = 00:00:00
BMC40102I MERGE INDEXSPACE STEP STARTING
BMC40795I FULL COPY DATASET IS MVSMAR3.LP.AFREX01.ICEXIP1.F003.G0008V00
BMC40799I MERGE STARTING FOR DATA SET DSNDXW.DSNDBD.AFREX01.ICEXIP1.ID001.A003
BMC40593I INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00
BMC40593I SPIN COMPLETE, ELAPSED TIME = 00:00:00
BMC40902I TOTAL WAITS = 1, TOTAL WAIT TIME = 00:00:00
BMC40981I FULL IC MVSMAR3.LP.AFREX01.ICEXIP1.F003.G0008V00 READ WAITS = 0, WAIT TIME = 00:00:00
BMC40905I SPACE WRITE WAITS = 1, WAIT TIME = 00:00:00
BMC40985I NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 11, USED = 11
BMC40790I NUMBER OF PAGES READ FROM FULL COPY TOTAL = 11, USED = 11
BMC40900I MERGE INDEXSPACE STEP COMPLETE
BMC40593I MERGE PROCESS COMPLETE, ELAPSED TIME = 00:00:00
BMC40352I DDNAME SYS00108 (DSN DSNDXW.DSNDBD.AFREX01.ICEX1P1.I0001.A003) WAS SUCCESSFULLY UNALLOCATED
BMC40352I DDNAME SYS00114 (DSN MVSMAR3.LP.AFREX01.ICEX1P1.F004.G0008V00) WAS SUCCESSFULLY UNALLOCATED
BMC40060I PENDING FLAG RBDP SET FOR
BMC40345I SPACE = AFREX01.ICEXP1
BMC40305I DSNUM = 4
BMC40091I 01/03/2011 18:32:09 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE
BMC40012I MERGE PHASE COMPLETE, ELAPSED TIME = 00:00:00, TIME SINCE UTILITY START = 00:00:11
BMC40868I MERGE PHASE COMPLETE, ACCUMULATED TCB TIME = 00:00:00
BMC40016I MERGE PHASE STARTING 01/03/2011 18:32:09
BMC40303I INDEX = AFR.ICEXP1
BMC40305I DSNUM = 4
BMC40350I DATA SET MVSMAR3.LP.AFREX01.ICEXIP1.F004.G0008V00 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00110 ELAPSED = 00:00:00
BMC40601I PENDING FLAG RBDP SET FOR
BMC40345I SPACE = AFREX01.ICEXIP1
BMC40305I DSNUM = 4
BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.ICEXP1.10001.A004
BMC40356I AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.ICEXP1.10001.A004
BMC40350I DATA SET DSNDXW.DSNDBD.AFREX01.ICEXIP1.10001.A004 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00114 ELAPSED = 00:00:00
BMC6304I THIS PHASE WILL RUN IN EXECUTION SUBTASK 1
BMC40593I START MERGE COMPLETE, ELAPSED TIME = 00:00:00
BMC40102I MERGE INDEXSPACE STEP STARTING
BMC40795I FULL COPY DATASET IS MVSMAR3.LP.AFREX01.ICEXIP1.F004.G0008V00
BMC40799I MERGE STARTING FOR DATA SET DSNDXW.DSNDBD.AFREX01.ICEXIP1.ID001.A004
BMC40593I INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00
BMC40593I SPIN COMPLETE, ELAPSED TIME = 00:00:00
BMC40902I TOTAL WAITS = 2, TOTAL WAIT TIME = 00:00:00
BMC40981I FULL IC MVSMAR3.LP.AFREX01.ICEXIP1.F004.G0008V00 READ WAITS = 1, WAIT TIME = 00:00:00
BMC40905I SPACE WRITE WAITS = 1, WAIT TIME = 00:00:00
BMC40985I NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 11, USED = 11
BMC40790I NUMBER OF PAGES READ FROM FULL COPY TOTAL = 11, USED = 11
BMC40900I MERGE INDEXSPACE STEP COMPLETE
BMC40593I MERGE PROCESS COMPLETE, ELAPSED TIME = 00:00:00
BMC40352I DDNAME SYS00114 (DSN DSNDXW.DSNDBD.AFREX01.ICEX1P1.I0001.A003) WAS SUCCESSFULLY UNALLOCATED
BMC40352I DDNAME SYS00110 (DSN MVSMAR3.LP.AFREX01.ICEX1P1.F004.G0008V00) WAS SUCCESSFULLY UNALLOCATED
BMC40060I PENDING FLAG RBDP SET FOR
BMC40345I SPACE = AFREX01.ICEXP1
BMC40305I DSNUM = 4
BMC40091I 01/03/2011 18:32:10 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE
BMC40012I MERGE PHASE COMPLETE, ELAPSED TIME = 00:00:00, TIME SINCE UTILITY START = 00:00:12
BMC40868I MERGE PHASE COMPLETE, ACCUMULATED TCB TIME = 00:00:00
**Example 2B: Using RECOVER INDEX**

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40016I</td>
<td>MERGE PHASE STARTING 01/03/2011 18:32:23</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 23</td>
</tr>
<tr>
<td>BMC40484I</td>
<td>MESSAGES FROM PHASE SUBTASK:</td>
</tr>
<tr>
<td>BMC40060I</td>
<td>PENDING FLAG RECP SET FOR</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 23</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDBD.AGREX01.TSEX1P1.10001.A023</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDBD.AGREX01.TSEX1P1.10001.A023</td>
</tr>
<tr>
<td>BMC40350I</td>
<td>DATA SET DSNDBD.AGREX01.TSEX1P1.10001.A023 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00179 ELAPSED = 00:00:00</td>
</tr>
<tr>
<td>BMC40485I</td>
<td>END OF PHASE SUBTASK MESSAGES</td>
</tr>
<tr>
<td>BMC6304I</td>
<td>THIS PHASE WILL RUN IN EXECUTION SUBTASK 1</td>
</tr>
<tr>
<td>BMC40593I</td>
<td>START MERGE COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40102I</td>
<td>MERGE TABLESPACE STEP STARTING</td>
</tr>
<tr>
<td>BMC40795I</td>
<td>FULL COPY DATASET IS MVSMAR3.LP.AGREX01.TSEX1P1.F000.G0008V00</td>
</tr>
<tr>
<td>BMC40799I</td>
<td>MERGE STARTING FOR DATA SET DSNDBD.AGREX01.TSEX1P1.10001.A023</td>
</tr>
<tr>
<td>BMC40593I</td>
<td>INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40593I</td>
<td>SPIN COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40902I</td>
<td>TOTAL WAITS = 1, TOTAL WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40981I</td>
<td>FULL IC MVSMAR3.LP.AGREX01.TSEX1P1.F000.G0008V00 READ WAITS = 8, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40905I</td>
<td>SPACE WRITE WAITS = 1, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40985I</td>
<td>NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 27, USED = 27</td>
</tr>
<tr>
<td>BMC40990I</td>
<td>NUMBER OF PAGES OUTPUT IS = 27</td>
</tr>
<tr>
<td>BMC40900I</td>
<td>MERGE TABLESPACE STEP COMPLETE</td>
</tr>
<tr>
<td>BMC40593I</td>
<td>MERGE PROCESS COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40352I</td>
<td>DDNAME SYS00179 (DSN DSNDBD.AGREX01.TSEX1P1.10001.A023) WAS SUCCESSFULLY UNALLOCATED</td>
</tr>
<tr>
<td>BMC40060I</td>
<td>PENDING FLAG RECP RESET FOR</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 23</td>
</tr>
<tr>
<td>BMC40016I</td>
<td>MERGE PHASE STARTING 01/03/2011 18:32:24</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40484I</td>
<td>MESSAGES FROM PHASE SUBTASK:</td>
</tr>
<tr>
<td>BMC40060I</td>
<td>PENDING FLAG RECP SET FOR</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDBD.AGREX01.TSEX1P1.10001.A024</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDBD.AGREX01.TSEX1P1.10001.A024</td>
</tr>
<tr>
<td>BMC40350I</td>
<td>DATA SET DSNDBD.AGREX01.TSEX1P1.10001.A024 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00184 ELAPSED = 00:00:00</td>
</tr>
<tr>
<td>BMC40485I</td>
<td>END OF PHASE SUBTASK MESSAGES</td>
</tr>
<tr>
<td>BMC6304I</td>
<td>THIS PHASE WILL RUN IN EXECUTION SUBTASK 1</td>
</tr>
<tr>
<td>BMC40593I</td>
<td>START MERGE COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40102I</td>
<td>MERGE TABLESPACE STEP STARTING</td>
</tr>
<tr>
<td>BMC40795I</td>
<td>FULL COPY DATASET IS MVSMAR3.LP.AGREX01.TSEX1P1.F000.G0008V00</td>
</tr>
<tr>
<td>BMC40799I</td>
<td>MERGE STARTING FOR DATA SET DSNDBD.AGREX01.TSEX1P1.10001.A024</td>
</tr>
<tr>
<td>BMC40593I</td>
<td>INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40593I</td>
<td>SPIN COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40902I</td>
<td>TOTAL WAITS = 1, TOTAL WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40981I</td>
<td>FULL IC MVSMAR3.LP.AGREX01.TSEX1P1.F000.G0008V00 READ WAITS = 8, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40905I</td>
<td>SPACE WRITE WAITS = 1, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40985I</td>
<td>NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 27, USED = 27</td>
</tr>
</tbody>
</table>
Figure 63  Example 2B—Edited AFPROPRINT for a simple RECOVER INDEX (part 6 of 6)

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40791I</td>
<td>NUMBER OF PAGES READ FROM FULL COPY TOTAL = 648, USED = 648</td>
</tr>
<tr>
<td>BMC40900I</td>
<td>MERGE TABLESPACE STEP COMPLETE</td>
</tr>
<tr>
<td>BMC40931I</td>
<td>MERGE PROCESS COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40521I</td>
<td>DNAME SYS001B1 (DSN DSNDMW.DSNDBD.AFREX01.TSEX1P1.TSEX1P1.10001.A024) WAS SUCCESSFULLY UNALLOCATED</td>
</tr>
<tr>
<td>BMC40521I</td>
<td>DNAME SYS000DB3 (DSN MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00) WAS SUCCESSFULLY UNALLOCATED</td>
</tr>
<tr>
<td>BMC40406I</td>
<td>PENDING FLAG RECP RESET FOR</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40091I</td>
<td>01/03/2011 18:32:24 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE</td>
</tr>
<tr>
<td>BMC40012I</td>
<td>MERGE PHASE COMPLETE. ELAPSED TIME = 00:00:00, TIME SINCE UTILITY START = 00:00:26</td>
</tr>
<tr>
<td>BMC40868I</td>
<td>MERGE PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:01</td>
</tr>
</tbody>
</table>

Figure 64  Edited AFROSUM for a simple RECOVER INDEX (part 1 of 4)

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40018I</td>
<td>RECOVER PLUS FOR DB2 V10.1.00</td>
</tr>
<tr>
<td>BMC40741I</td>
<td>COPYRIGHT BMC SOFTWARE INC. 1991-2011</td>
</tr>
<tr>
<td>BMC40791I</td>
<td>RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884</td>
</tr>
<tr>
<td>BMC40741I</td>
<td>CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO <a href="mailto:SUPPORT@BMC.COM">SUPPORT@BMC.COM</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40791I</td>
<td>OBJECT SUMMARY:</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 1 OF 24</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = X'021C'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = SYSDEFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40820I</td>
<td>TABLESPACE RECOVERY</td>
</tr>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'C4BC51210168' TO LOGPOINT = X'C4BC5147226E'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40841I</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>2011-01-03 18.31.30.4466 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40641I</td>
<td>2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC5141D87E' QUIESC UTILITY POINT - WRITE(YES)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40791I</td>
<td>OBJECT SUMMARY:</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 2 OF 24</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = X'021C'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = SYSDEFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40820I</td>
<td>TABLESPACE RECOVERY</td>
</tr>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'C4BC51210168' TO LOGPOINT = X'C4BC5147226E'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40841I</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>2011-01-03 18.31.30.4466 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40641I</td>
<td>2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC5141D87E' QUIESC UTILITY POINT - WRITE(YES)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40791I</td>
<td>OBJECT SUMMARY:</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 3 OF 24</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = X'021C'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = SYSDEFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
</tbody>
</table>
### Figure 64  Edited AFROSUM for a simple RECOVER INDEX (part 2 of 4)

<table>
<thead>
<tr>
<th>BMC40723</th>
<th>ENCODING SCHEME = EBCDIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40820</td>
<td>TABLESPACE RECOVERY</td>
</tr>
<tr>
<td>BMC40827</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585</td>
<td>FROM LOGPOINT = X'C4BC51210168' TO LOGPOINT = X'C4BC5147226E'</td>
</tr>
<tr>
<td>BMC40840</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC40640</td>
<td>2011-01-03 18.31.30.4466 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40302</td>
<td>DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00</td>
</tr>
<tr>
<td>BMC40641</td>
<td>SHRLEVEL = CHANGE SITETYPE = LP DSNUM = 0</td>
</tr>
<tr>
<td>BMC40640</td>
<td>2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC5141D87E' QUIESCE UTILITY POINT - WRITE(YES)</td>
</tr>
<tr>
<td>BMC40709</td>
<td>OBJECT SUMMARY:</td>
</tr>
<tr>
<td>BMC40301</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40713</td>
<td>PARTITION = 4 OF 24</td>
</tr>
<tr>
<td>BMC40711</td>
<td>OBID = X'02IC' PSID = X'000E'</td>
</tr>
<tr>
<td>BMC40712</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716</td>
<td>STOGROUP = SYSDEFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
<tr>
<td>BMC40723</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

| BMC40820 | TABLESPACE RECOVERY      |
| BMC40827 | LOG RESOURCE RANGE:      |
| BMC40585 | FROM LOGPOINT = X'C4BC51210168' TO LOGPOINT = X'C4BC5147226E' |
| BMC40840 | INPUT COPY / SYSCOPY DATA: |
| BMC40640 | 2011-01-03 18.31.30.4466 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY |
| BMC40302 | DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00 |
| BMC40641 | SHRLEVEL = CHANGE SITETYPE = LP DSNUM = 0 |
| BMC40640 | 2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC5141D87E' QUIESCE UTILITY POINT - WRITE(YES) |
| BMC40709 | OBJECT SUMMARY: |
| BMC40301 | TABLESPACE = AFREX01.TSEX1P1 |
| BMC40713 | PARTITION = 5 OF 24 |
| BMC40711 | OBID = X'02IC' PSID = X'000E' |
| BMC40712 | ORIGINAL STATUS = RW |
| BMC40716 | STOGROUP = SYSDEFLT PRIQTY = 25 SECQTY = 13 |
| BMC40723 | ENCODING SCHEME = EBCDIC |

| BMC40820 | TABLESPACE RECOVERY      |
| BMC40827 | LOG RESOURCE RANGE:      |
| BMC40585 | FROM LOGPOINT = X'C4BC51210168' TO LOGPOINT = X'C4BC5147226E' |
| BMC40840 | INPUT COPY / SYSCOPY DATA: |
| BMC40640 | 2011-01-03 18.31.30.4466 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY |
| BMC40302 | DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00 |
| BMC40641 | SHRLEVEL = CHANGE SITETYPE = LP DSNUM = 0 |
| BMC40640 | 2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC5141D87E' QUIESCE UTILITY POINT - WRITE(YES) |
| BMC40709 | OBJECT SUMMARY: |
| BMC40301 | TABLESPACE = AFREX01.TSEX1P1 |
| BMC40713 | PARTITION = 6 OF 24 |
| BMC40711 | OBID = X'02IC' PSID = X'000E' |
| BMC40712 | ORIGINAL STATUS = RW |
| BMC40716 | STOGROUP = SYSDEFLT PRIQTY = 25 SECQTY = 13 |
| BMC40723 | ENCODING SCHEME = EBCDIC |

| BMC40820 | TABLESPACE RECOVERY      |
| BMC40827 | LOG RESOURCE RANGE:      |
| BMC40585 | FROM LOGPOINT = X'C4BC51210168' TO LOGPOINT = X'C4BC5147226E' |
| BMC40840 | INPUT COPY / SYSCOPY DATA: |
| BMC40640 | 2011-01-03 18.31.30.4466 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY |
| BMC40302 | DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00 |
| BMC40641 | SHRLEVEL = CHANGE SITETYPE = LP DSNUM = 0 |
| BMC40640 | 2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC5141D87E' QUIESCE UTILITY POINT - WRITE(YES) |
| BMC40709 | OBJECT SUMMARY: |
| BMC40301 | TABLESPACE = AFREX01.TSEX1P1 |
| BMC40713 | PARTITION = 7 OF 24 |
| BMC40711 | OBID = X'02IC' PSID = X'000E' |
| BMC40712 | ORIGINAL STATUS = RW |
| BMC40716 | STOGROUP = SYSDEFLT PRIQTY = 25 SECQTY = 13 |
| BMC40723 | ENCODING SCHEME = EBCDIC |
Figure 64  Edited AFROSUM for a simple RECOVER INDEX (part 3 of 4)

<table>
<thead>
<tr>
<th>BMC406411</th>
<th>SHRLEVEL = CHANGE  SITETYPE = LP DSNUM = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC406401</td>
<td>2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC514187E' QUIESCE UTILITY POINT - WRITE(YES)</td>
</tr>
<tr>
<td>BMC407091</td>
<td>OBJECT SUMMARY:</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC407131</td>
<td>PARTITION = 7 OF 24</td>
</tr>
<tr>
<td>BMC407111</td>
<td>DBID = X'021C'  PSID = X'00DE'</td>
</tr>
<tr>
<td>BMC407121</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC407161</td>
<td>STOGROUP = SYSDFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
<tr>
<td>BMC407231</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
<tr>
<td>BMC408201</td>
<td>TABLESPACE RECOVERY</td>
</tr>
<tr>
<td>BMC408271</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC405851</td>
<td>FROM LOGPOINT = X'C4BC51210168'  TO LOGPOINT = X'C4BC5147226E'</td>
</tr>
<tr>
<td>BMC408401</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC406401</td>
<td>2011-01-03 18.31.30.4466 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC403021</td>
<td>DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00</td>
</tr>
<tr>
<td>BMC406411</td>
<td>SHRLEVEL = CHANGE SITETYPE = LP DSNUM = 0</td>
</tr>
<tr>
<td>BMC406401</td>
<td>2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC514187E' QUIESCE UTILITY POINT - WRITE(YES)</td>
</tr>
<tr>
<td>BMC407091</td>
<td>OBJECT SUMMARY:</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC407131</td>
<td>PARTITION = 8 OF 24</td>
</tr>
<tr>
<td>BMC407111</td>
<td>DBID = X'021C'  PSID = X'00DE'</td>
</tr>
<tr>
<td>BMC407121</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC407161</td>
<td>STOGROUP = SYSDFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
<tr>
<td>BMC407231</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
<tr>
<td>BMC408201</td>
<td>TABLESPACE RECOVERY</td>
</tr>
<tr>
<td>BMC408271</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC405851</td>
<td>FROM LOGPOINT = X'C4BC51210168'  TO LOGPOINT = X'C4BC5147226E'</td>
</tr>
<tr>
<td>BMC408401</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC406401</td>
<td>2011-01-03 18.31.30.4466 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC403021</td>
<td>DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00</td>
</tr>
<tr>
<td>BMC406411</td>
<td>SHRLEVEL = CHANGE SITETYPE = LP DSNUM = 0</td>
</tr>
<tr>
<td>BMC406401</td>
<td>2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC514187E' QUIESCE UTILITY POINT - WRITE(YES)</td>
</tr>
<tr>
<td>BMC407091</td>
<td>OBJECT SUMMARY:</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC407131</td>
<td>PARTITION = 9 OF 24</td>
</tr>
<tr>
<td>BMC407111</td>
<td>DBID = X'021C'  PSID = X'00DE'</td>
</tr>
<tr>
<td>BMC407121</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC407161</td>
<td>STOGROUP = SYSDFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
<tr>
<td>BMC407231</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
<tr>
<td>BMC408201</td>
<td>TABLESPACE RECOVERY</td>
</tr>
<tr>
<td>BMC408271</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC405851</td>
<td>FROM LOGPOINT = X'C4BC51210168'  TO LOGPOINT = X'C4BC5147226E'</td>
</tr>
<tr>
<td>BMC408401</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC406401</td>
<td>2011-01-03 18.31.30.4466 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC403021</td>
<td>DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00</td>
</tr>
<tr>
<td>BMC406411</td>
<td>SHRLEVEL = CHANGE SITETYPE = LP DSNUM = 0</td>
</tr>
<tr>
<td>BMC406401</td>
<td>2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC514187E' QUIESCE UTILITY POINT - WRITE(YES)</td>
</tr>
<tr>
<td>BMC407091</td>
<td>OBJECT SUMMARY:</td>
</tr>
<tr>
<td>BMC403011</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC407131</td>
<td>PARTITION = 10 OF 24</td>
</tr>
<tr>
<td>BMC407111</td>
<td>DBID = X'021C'  PSID = X'00DE'</td>
</tr>
<tr>
<td>BMC407121</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC407161</td>
<td>STOGROUP = SYSDFLT PRIQTY = 25 SECQTY = 13</td>
</tr>
<tr>
<td>BMC407231</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>
**Example 2B: Using RECOVER INDEX**

**Figure 64  Edited AFROSUM for a simple RECOVER INDEX (part 4 of 4)**

<table>
<thead>
<tr>
<th>BMC40820I</th>
<th>TABLESPACE RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'C4BC51210168' TO LOGPOINT = X'C4BC5147226E'</td>
</tr>
<tr>
<td>BMC40840I</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>2011-01-03 18.31.40.9804 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'C4BC51210168' TO LOGPOINT = X'C4BC5147226E'</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>2011-01-03 18.31.30.4466 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMAR3.LP.AFREX01.TSEXP1.F003.00008V00</td>
</tr>
<tr>
<td>BMC40641I</td>
<td>SHRLEVEL = CHANGE SITETYPE = LP DSNUM = 0</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC5141D87E' QUIESCE UTILITY POINT - WRITE(YES)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40709I</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40461I</td>
<td>INDEXSPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 23 OF 24</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = X'021C' PSID = X'0027'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = AFR.TSEX1P1</td>
</tr>
<tr>
<td>BMC96160I</td>
<td>PAGE SIZE = 4K</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = SYSDEFLT PRIQTY = 25 SECRYPT = 25</td>
</tr>
<tr>
<td>BMC40830I</td>
<td>ON TABLE AFR.TSEX1P (OBID X'000F') IN TABLESPACE AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40836I</td>
<td>TABLE CARDINALITY = 24000</td>
</tr>
<tr>
<td>BMC40832I</td>
<td>NON-CLUSTERING INDEX</td>
</tr>
<tr>
<td>BMC40834I</td>
<td>NON-UNIQUE INDEX</td>
</tr>
<tr>
<td>BMC40844I</td>
<td>TYPE 2</td>
</tr>
<tr>
<td>BMC40846I</td>
<td>COPY YES COPYLRSN = X'C4BC500B8200'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40843I</th>
<th>INDEX RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'C4BC51210168' TO LOGPOINT = X'C4BC5147226E'</td>
</tr>
<tr>
<td>BMC40840I</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>2011-01-03 18.31.40.2688 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F023.00008V00</td>
</tr>
<tr>
<td>BMC40641I</td>
<td>SHRLEVEL = CHANGE SITETYPE = LP DSNUM = 23</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC5141D87E' QUIESCE UTILITY POINT - WRITE(YES)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40709I</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40461I</td>
<td>INDEXSPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 24 OF 24</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = X'021C' PSID = X'0027'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = AFR.TSEX1P1</td>
</tr>
<tr>
<td>BMC96160I</td>
<td>PAGE SIZE = 4K</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = SYSDEFLT PRIQTY = 25 SECRYPT = 25</td>
</tr>
<tr>
<td>BMC40830I</td>
<td>ON TABLE AFR.TSEX1P (OBID X'000F') IN TABLESPACE AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40836I</td>
<td>TABLE CARDINALITY = 24000</td>
</tr>
<tr>
<td>BMC40832I</td>
<td>NON-CLUSTERING INDEX</td>
</tr>
<tr>
<td>BMC40834I</td>
<td>NON-UNIQUE INDEX</td>
</tr>
<tr>
<td>BMC40844I</td>
<td>TYPE 2</td>
</tr>
<tr>
<td>BMC40846I</td>
<td>COPY YES COPYLRSN = X'C4BC500B8200'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40843I</th>
<th>INDEX RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'C4BC51210168' TO LOGPOINT = X'C4BC5147226E'</td>
</tr>
<tr>
<td>BMC40840I</td>
<td>INPUT COPY / SYSCOPY DATA:</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>2011-01-03 18.31.40.9804 RBA/LRSN X'C4BC51210168' FULL IMAGE COPY</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F023.00008V00</td>
</tr>
<tr>
<td>BMC40641I</td>
<td>SHRLEVEL = CHANGE SITETYPE = LP DSNUM = 23</td>
</tr>
<tr>
<td>BMC40640I</td>
<td>2011-01-03 18.31.54.6480 RBA/LRSN X'C4BC5141D87E' QUIESCE UTILITY POINT - WRITE(YES)</td>
</tr>
</tbody>
</table>
### Example 2B: Using RECOVER INDEX

#### Figure 65  Example 2B—Edited AFRPLAN for a simple RECOVER INDEX (part 1 of 6)

<table>
<thead>
<tr>
<th>BMC40018I</th>
<th>RECOVER PLUS for DB2 V10.1.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40473I</td>
<td>COPYRIGHT BMC SOFTWARE INC. 1991-2011</td>
</tr>
<tr>
<td>BMC40474I</td>
<td>CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO <a href="mailto:SUPPORT@BMC.COM">SUPPORT@BMC.COM</a></td>
</tr>
</tbody>
</table>

#### BMC40700I EXECUTION PLAN SUMMARY:

<table>
<thead>
<tr>
<th>BMC40701I</th>
<th>PHASE: MERGE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BMC40721I</strong></td>
<td>PHASE INITIALIZATION:</td>
</tr>
<tr>
<td><strong>BMC40301I</strong></td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td><strong>BMC40305I</strong></td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE INPUT IMAGE COPY FILE</td>
</tr>
<tr>
<td><strong>BMC40302I</strong></td>
<td>DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE 2 INPUT FILE BUFFER(S)</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>OPEN IMAGE COPY FILE</td>
</tr>
<tr>
<td><strong>BMC40302I</strong></td>
<td>DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td><strong>BMC40345I</strong></td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td><strong>BMC40305I</strong></td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td><strong>BMC40345I</strong></td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td><strong>BMC40305I</strong></td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE 2 SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td><strong>BMC40702I</strong></td>
<td>STEP: MERGE SPACE</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td><strong>BMC40301I</strong></td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td><strong>BMC40305I</strong></td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td><strong>BMC40735I</strong></td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td><strong>BMC40345I</strong></td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td><strong>BMC40305I</strong></td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40701I</th>
<th>PHASE: MERGE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BMC40721I</strong></td>
<td>PHASE INITIALIZATION:</td>
</tr>
<tr>
<td><strong>BMC40301I</strong></td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td><strong>BMC40305I</strong></td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td><strong>BMC40345I</strong></td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td><strong>BMC40305I</strong></td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td><strong>BMC40345I</strong></td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td><strong>BMC40305I</strong></td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td><strong>BMC40738I</strong></td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td>Line</td>
<td>Command</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40702I</td>
<td>STEP: MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Line</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40701I</td>
<td>PHASE: MERGE</td>
<td></td>
</tr>
<tr>
<td>BMC40721I</td>
<td>PHASE INITIALIZATION:</td>
<td></td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
<td></td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CREATE SPACE</td>
<td></td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
<td></td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
<td></td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
<td></td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
<td></td>
</tr>
<tr>
<td>BMC40702I</td>
<td>STEP: MERGE SPACE</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
<td></td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
<td></td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
<td></td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
<td></td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
<td></td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Line</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40701I</td>
<td>PHASE: MERGE</td>
<td></td>
</tr>
<tr>
<td>BMC40721I</td>
<td>PHASE INITIALIZATION:</td>
<td></td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
<td></td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 4</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CREATE SPACE</td>
<td></td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
<td></td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 4</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
<td></td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
<td></td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 4</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
<td></td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
<td></td>
</tr>
</tbody>
</table>
Example 2B: Using RECOVER INDEX

Figure 65  Example 2B—Edited AFRPLAN for a simple RECOVER INDEX  (part 3 of 6)

<table>
<thead>
<tr>
<th>BMC40738I</th>
<th>ALLOCATE PASTE BUFFER (4096 BYTES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40702I</td>
<td>STEP:  MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 4</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 4</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC40701I</td>
<td>PHASE:  MERGE</td>
</tr>
<tr>
<td>BMC40721I</td>
<td>PHASE INITIALIZATION:</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 22</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 22</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 22</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40702I</td>
<td>STEP:  MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 22</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 22</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC40701I</td>
<td>PHASE:  MERGE</td>
</tr>
<tr>
<td>BMC40721I</td>
<td>PHASE INITIALIZATION:</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 23</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 23</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 23</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
</tbody>
</table>
### Figure 65  Example 2B—Edited AFRPLAN for a simple RECOVER INDEX (part 4 of 6)

<table>
<thead>
<tr>
<th>Line</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40702I</td>
<td>STEP: MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 23</td>
</tr>
<tr>
<td>BMC4035I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 23</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CLOSE IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 INPUT FILE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE INPUT IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMAR3.LP.AFREX01.TSEX1P1.F000.G0008V00</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>

### Example 2B: Using RECOVER INDEX

356  RECOVER PLUS for DB2 Reference Manual
**Example 2B: Using RECOVER INDEX**

**Chapter 5 Examples of RECOVER PLUS jobs**

**Figure 65**  Example 2B—Edited AFRPLAN for a simple RECOVER INDEX  (part 5 of 6)

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 INPUT FILE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>OPEN IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMA3.LP.AFREX01.INEX1P1.F001.G0008V00</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>SET PAGE SET REBUILD PENDING (PSRBD) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.INEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.INEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40702I</td>
<td>STEP: MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = AFR.INEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.INEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CLOSE IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMA3.LP.AFREX01.INEX1P1.F001.G0008V00</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 INPUT FILE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE INPUT IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMA3.LP.AFREX01.INEX1P1.F001.G0008V00</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET PAGE SET REBUILD PENDING (PSRBD) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
<tr>
<td><strong>BMC40701I</strong> PHASE: MERGE</td>
<td></td>
</tr>
<tr>
<td><strong>BMC40721I</strong> PHASE INITIALIZATION:</td>
<td></td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = AFR.ICEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE INPUT IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMA3.LP.AFREX01.ICEX1P1.F024.G0008V00</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 INPUT FILE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>OPEN IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMA3.LP.AFREX01.ICEX1P1.F024.G0008V00</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET REBUILD PENDING (RBPD) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.ICEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.ICEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
</tr>
</tbody>
</table>

*Note: This example demonstrates the editing process for AFRPLAN, focusing on the allocation and management of buffers and space during the RECOVER INDEX operation.*
Example 2B: Using RECOVER INDEX

Figure 65  Example 2B—Edited AFRPLAN for a simple RECOVER INDEX (part 6 of 6)

| BMC40738I | ALLOCATE PAGE CHECK BUFFER (4096 BYTES) |
| BMC40738I | FILE OPTIMIZATION |
| BMC40738I | ALLOCATE 2 SPACE OUTPUT BUFFER(S) |
| BMC40738I | ALLOCATE SPACEMAP BUFFER (4096 BYTES) |
| BMC40738I | ALLOCATE PASTE BUFFER (4096 BYTES) |
| BMC40702I | STEP: MERGE SPACE |
| BMC40738I | MERGE SPACE |
| BMC40303I | INDEX = AFR.ICEX1P1 |
| BMC40305I | DSNUM = 24 |
| BMC40735I | PHASE TERMINATION: |
| BMC40738I | DISCONNECT FROM SPACE |
| BMC40738I | DEALLOCATE SPACE |
| BMC40345I | SPACE = AFREX01.ICEX1P1 |
| BMC40305I | DSNUM = 24 |
| BMC40738I | FREE PAGE CHECK BUFFER |
| BMC40738I | FREE SPACEMAP BUFFER |
| BMC40738I | FREE PASTE BUFFER |
| BMC40738I | FREE 2 OUTPUT SPACE BUFFER(S) |
| BMC40738I | FREE PAGE CHECK BUFFER |
| BMC40738I | FREE PASTE BUFFER |
| BMC40738I | FREE SPACEMAP BUFFER |
| BMC40738I | CLOSE IMAGE COPY FILE |
| BMC40302I | DSN = MVSMAR3.LP.AFREX01.ICEX1P1.F024.G0008V00 |
| BMC40738I | FREE 2 INPUT FILE BUFFER(S) |
| BMC40738I | DEALLOCATE INPUT IMAGE COPY FILE |
| BMC40302I | DSN = MVSMAR3.LP.AFREX01.ICEX1P1.F024.G0008V00 |
| BMC40738I | RESET REBUILD PENDING (RBDP) FLAG |
| BMC40738I | RECORD RESTART INFORMATION |
Example 3: Creating and dynamically allocating output copies

In this example (Figure 66), RECOVER PLUS recovers one partition of the table space and creates one image copy. RECOVER PLUS uses dynamic allocation for the image copy.

The OUTPUT command is used to dynamically allocate the copy that is made.

Figure 66  Example 3—JCL for recovering one partition and creating an image copy

```
//AFREX03 JOB (PARM),'EXAMPLE 3',
//       CLASS=Q,NOTIFY=&SYSUID,
//       MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN,REGION=0M,
//       PARM='DGE,EXAMPLE03,NEW/RESTART,MSGLEVEL(2)'   //STEPLIB DD DISP=SHR,DSN=product.libraries
//       DD DISP=SHR,DSN=DB2.DSNEXIT
//       DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN     DD *
OUTPUT CP00001
   DSNNAME RDASTC.BMCDBSMP.BMCTS011.COPYDS3(+1)
   UNIT SYSDA
   MODELDCB SYS1.MODEL
   SPACE (20,1) CYL
RECOVER TABLESPACE BMCDBSMP.BMCTS011 DSNUM 03
   OUTCOPY YES
   OUTCOPYDDN CP00001
/*
```
Example 4: Recovering to a specified copy

With the TOCOPY option, you can specify the name of the last image copy to be used by RECOVER PLUS when restoring the table space. If the copy that you specify is an incremental image copy, the prior full copy and any intervening incremental image copies are merged.

Example 4A: Using TOCOPY with a named copy

The example shown in Figure 67 illustrates a table space recovery using only a full image copy and an index rebuild that includes all of the indexes on the table space.

Figure 67 Example 4A—JCL using TOCOPY with a named copy

```plaintext
//AFREX04A JOB (PAFR),'EXAMPLE 4A'.
//              CLASS=Q,NOTIFY=&SYSUID.
//              MSGCLASS=X
///*
//RECOVER EXEC PGM=AFRMAIN,REGION=0M.
//              PARM='DGE.EXAMPLEE04A,NEW/RESTART,MSGLEVEL(2)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
//              DD DISP=SHR,DSN=DB2.DSNEXIT
//              DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN     DD *
RECOVER TABLESPACE BMCDBSMP.BMCTS014
TOCOPY RDASTC.BMCDBSMP.BMCTS014.IC(0)
REBUILD INDEX(ALL) TABLESPACE BMCDBSMP.BMCTS014 NOWORKDDN
/*
//
```

Example 4B: Using TOCOPY LASTCOPY

The example shown in Figure 68 on page 361 uses the keyword LASTCOPY for the value of the TOCOPY data set name instead of providing the name of the copy. Using TOCOPY LASTCOPY can be useful in a number of routine situations. In the example, a routine backup of three table spaces is performed (using COPY PLUS) followed by a batch application that updates those table spaces with transactions that have accumulated since the prior backup. This process is followed by a conditional recovery.

If the batch application:

- ends abnormally, the spaces are restored to the state existing at the time the current copy was made using TOCOPY LASTCOPY instead of naming the copy
- runs successfully, no recovery is performed.
Using TOCOPY LASTCOPY in this way avoids having to modify the JCL for routine operations.

**Figure 68  Example 4B—JCL using TOCOPY LASTCOPY (part 1 of 2)**

```plaintext
//AFREX04B JOB (PFTR), 'EXAMPLE 4B',
// CLASS=Q, NOTIFY=YESUID,
// MSGCLASS=X
/*
* -------------------------------
* THIS STEP TAKES FULL IMAGE COPIES OF THREE TABLE SPACES
* -------------------------------
*/
/*
*/
//BMCCOPY EXEC PGM=ACPMAIN, REGION=0M,
// PARM='DGECOPY4B.NEW/RESET,MSGLEVEL(1)'
// STEPLIB DD DISP=SHR,DSN=productlibraries
// DD DISP=SHR, DSN=DB2.DSNEXIT
// DD DISP=SHR, DSN=DB2.DSNLOAD
// CP00001 DD DSN=RDASTC.BMCDBSMP.BMCTS001.IC(+1), DISP=(,CATLG),
// DCB=(SYS1.MODEL), UNIT=SYSDA, SPACE=(CYL,(25,5))
// CP00002 DD DSN=RDASTC.BMCDBSMP.BMCTS002.IC(+1), DISP=(,CATLG),
// DCB=(SYS1.MODEL), UNIT=SYSDA, SPACE=(CYL,(25,5))
// CP00003 DD DSN=RDASTC.BMCDBSMP.BMCTS003.IC(+1), DISP=(,CATLG),
// DCB=(SYS1.MODEL), UNIT=SYSDA, SPACE=(CYL,(25,5))
// SYSPRINT DD SYSOUT=* 
*/
/*
* -------------------------------
* THIS STEP UPDATES THE TABLE SPACES COPIES IN THE PREVIOUS STEP
* -------------------------------
*/
/*
* UPDATER EXEC PGM=................................................
* ...................................................................
* ... (BATCH APPLICATION THAT UPDATES THE COPIED TABLE SPACES) ......
* ... (SIMULATED WITH FOLLOWING SQL) .............................. ......
*/
/* UPDATE BMCTB001, BMCTB002, BMCTB003 
*==================================================================
*/
```
Example 4B: Using TOCOPY LASTCOPY

Figure 68  Example 4B—JCL using TOCOPY LASTCOPY (part 2 of 2)
Example 5: Recovering to a specific log point

This example illustrates two ways of recovering all parts of a table space and its indexes to a prior point in time by specifying TOLOGPOINT with the keyword LASTQUIESCE. This method allows a recovery to the last quiesce point registered for the table space in SYSIBM.SYSCOPY.

The log point specified by TOLOGPOINT LASTQUIESCE defines the point at which recovery stops. Only log records with starting log points equal to or lower than LASTQUIESCE and only image copies with log points less than LASTQUIESCE are used in the recovery.

Example 5A: Using the TOLOGPOINT option with RECOVER

Example 5A (Figure 69) uses image copies and archive and active logs to recover all partitions of the table space. The indexes are also recovered from image copies and logs. A SYSPICK DD statement is included to obtain a list of all of the input tape and cartridge volumes that are allocated during recovery.

Figure 69   Example 5A—JCL using TOLOGPOINT with RECOVER

```
//AFREX05A JOB (PAFR), 'EXAMPLE 5A',
//         CLASS=Q, NOTIFY=SYSUID,
//         MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN, REGION=0M,
//         PARM='DGE, EXAMPLE5A, NEW/RESTART, MSGLEVEL(2)'
//STEPLIB DD DISP=SHR, DSN=product.libraries
//         DD DISP=SHR, DSN=DB2.DSNEXIT
//         DD DISP=SHR, DSN=DB2.DSNLOAD
//SYSIN     DD *
//OPTIONS INDEXLOG YES
RECOVER TABLESPACE BMCDBSMP.BMCTS012
TOLOGPOINT LASTQUIESCE
RECOVER INDEX(ALL) TABLESPACE BMCDBSMP.BMCTS012
/*
//SYSPICK DD SYSOUT=* //
```
Example 5B: Using TOLOGPOINT and BACKOUT

Example 5B (Figure 70) uses the BACKOUT feature to recover the table space and its indexes.

Figure 70 Example 5B—JCL using TOLOGPOINT and BACKOUT

```
//AFREX05B JOB (PAFR), 'EXAMPLE 5B',
//         CLASS=Q, NOTIFY=&SYSUID,
//         MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN, REGION=OM,
//         PARM='DGE,EXAMPLE5B,NEW/RESTART,MSGLEVEL(2)'
//STEPLIB DD DISP=SHR, DSN=product.libries
//        DD DISP=SHR, DSN=DB2.DSNEXIT
//        DD DISP=SHR, DSN=DB2.DSNLOAD
//SYSIN     DD *
//OPTIONS INDEXLOG YES
RECOVER TABLESPACE BMCDBSMP.BMCTS013
   TOLOGPOINT LASTQUIESC
   BACKOUT
RECOVER INDEX(ALL) TABLESPACE BMCDBSMP.BMCTS013
//```
Example 6: Extracting nonpartitioned index keys

This example (Figure 71) illustrates the recovery of parts 1 and 2 of a 6 partition table space, the recovery of parts 1 and 2 of its partitioned index, and the unloading of keys from those partitions for the nonpartitioned index. The unloaded keys are sorted before being written to the work file. Notice that there is only one work file in this job and that the DDNAME for the work file defaults to SKEY. The keys for partitions 3 through 6 are unloaded and sorted in two other similar jobs, which can run concurrently with this one. The output from these jobs is used in “Example 7: Building a nonpartitioned index” on page 366 to build an index.

Figure 71   Example 6—JCL recovering a partitioned table space and partitioned index and unloading keys for a nonpartitioned index

```plaintext
//AFREX06 JOB (0000), 'EXAMPLE 6',
//        CLASS=Q, NOTIFY=&SYSUID,
//        MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN, REGION=0M,
//            PARM='DGE, EXAMPLE6, NEW/RESTART, MSGLEVEL(2)'
//STEPLIB DD DISP=SHR, DSN=product. libraries
//        DD DISP=SHR, DSN=DB2. DSNEXIT
//        DD DISP=SHR, DSN=DB2. DSNLOAD
//SKEY DD DSN=RDASTC. BMDCBJ. PARTS12. SUNKEY1,
//        DISP=(NEW, CATLG, DELETE),
//        UNIT=SYSDA,
//        SPACE=(CYL, (1,1)),
//        DCB=(SYS1.MODEL)
//SYSIN DD *
RECOVER TABLESPACE BMDCBJ. BMCTSJ DSNUM 01
   TABLESPACE BMDCBJ. BMCTSJ DSNUM 02 LOGSORT
REBUILD INDEX (RDASTC. BMCIXJ1) PART 01
REBUILD INDEX (RDASTC. BMCIXJ1) PART 02
RECOVER UNLOADKEYS (RDASTC. BMCIXJ2) PART 01
RECOVER UNLOADKEYS (RDASTC. BMCIXJ2) PART 02
/*
```
Example 7: Building a nonpartitioned index

This example (Figure 72) uses the keys that were unloaded and sorted in “Example 6: Extracting nonpartitioned index keys” on page 365 to build an index.

Figure 72  Example 7—JCL building an index with sorted unloaded keys

```
//AFREX07 JOB (0000),"EXAMPLE 7",
//        CLASS=Q,NOTIFY=ASYSUID
//        MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN,REGION=0M,
//              PARM='DGE,EXAMPLE07,NEW/RESTART,MSGLEVEL(2)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
//       DD DISP=SHR,DSN=DB2.DSNEXIT
//       DD DISP=SHR,DSN=DB2.DSNLOAD
//SKEY1 DD DISP=SHR,DSN=RDASTC.BMCDBJ.PARTS12.SUNKEY1
//SKEY2 DD DISP=SHR,DSN=RDASTC.BMCDBJ.PARTS34.SUNKEY2
//SKEY3 DD DISP=SHR,DSN=RDASTC.BMCDBJ.PARTS56.SUNKEY3
//SYSIN DD *
//RECOVER BUILDINDEX (RDASTC.BMCIXJ2)
/*
```
Example 8: Recovering using tape-stacked data sets

Example 8 consists of two jobs. The first job uses the COPY PLUS utility to stack three table spaces to different data sets on the same tape volume. The second job uses RECOVER PLUS to recover those table spaces.

*Tape stacking* refers to copying several small table spaces or table space partitions to different data sets on one or more tape volumes. The advantage of tape stacking is that it reduces the number of tapes needed for your image copies.

RECOVER PLUS dynamically allocates the data set and the tape volumes and avoids excessive rewinding and positioning. You do not need to specify the data sets in the JCL.

Example 8A: Making stacked image copies

The JCL in Figure 73 makes one full image copy of each of three table spaces, registers them, and writes them to the same cartridge tape.

**Figure 73  Example 8A—JCL for making stacked image copies**

```pl
//AFREX08A JOB (PAFR), 'EXAMPLE 8A',
//         CLASS=Q, NOTIFY=&SYSUID,
//         MSGCLASS=X
//*
//ACPCOPY EXEC PGM=ACPMAIN, REGION=0M,
//         PARM='DGE, EXAMPLE8A, NEW/RESET'
//STEPLIB DD DISP=SHR, DSN=product.libraries
// DD DISP=SHR, DSN=DB2.DSNEXIT
// DD DISP=SHR, DSN=DB2.DSNLOAD
//SYSIN DD *

OUTPUT STAKTAPE
UNIT CART
STACK YES
DSNAME &UID.&DB.&TS.D&DATE.T&TIME
MODELDCB SYS1.MODEL
COPY TABLESPACE BMCDBSMP.BMCTS001
   FULL YES
   COPYDDN(STAKTAPE)
COPY TABLESPACE BMCDBSMP.BMCTS002
   FULL YES
   COPYDDN(STAKTAPE)
COPY TABLESPACE BMCDBSMP.BMCTS003
   FULL YES
   COPYDDN(STAKTAPE)
/
//SYSPRINT DD SYSOUT=* 
//
```
Example 8B: Recovering with stacked image copies

The JCL in Figure 74 performs a recovery using the three image copies.

NOTE
In this example, you do not need to order the RECOVER TABLESPACE statements. RECOVER PLUS orders the activities to use the data sets sequentially.

Figure 74  Example 8B—JCL for recovering with stacked cataloged image copies as input

```plaintext
//AFREX08B JOB (PAFR), 'EXAMPLE 8B',
//         CLASS=Q, NOTIFY=&SYSUID,
//         MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN, REGION=5M,
//             PARM='DGE,EXAMPLE8B,NEW/RESTART,MSGLEVEL(2)'
//STEPLIB DD DISP=SHR, DSN=product.libraries
//         DD DISP=SHR, DSN=DB2.DSNEXIT
//        DD DISP=SHR, DSN=DB2.DSNLOAD
//SYSIN DD *
RECOVER TABLESPACE BMCDBSMP.BMCTS001 TOCOPY LASTCOPY
RECOVER TABLESPACE BMCDBSMP.BMCTS002 TOCOPY LASTCOPY
RECOVER TABLESPACE BMCDBSMP.BMCTS003 TOCOPY LASTCOPY
/*
//SYSPICK DD SYSOUT=* 
/*
```
Example 9: Recovering to a non-DB2 data set

Recovering to data sets not used by DB2 is useful for testing or for setting up recovered spaces to switch to while READ ONLY access continues to the real data sets.

Although INDEP INTABLESPACE is not explicitly specified with REBUILD INDEX in the SYSIN statement in the recovery JCL in Figure 75, RECOVER PLUS assumes it and defaults to the RECOVER TABLESPACE non-DB2 data set for index data input.

### Directing recovery output

When you use the RECOVER PLUS redirection feature to direct the output of a table space or index recovery to a data set other than the one normally used by DB2 or to extract index information from such a data set, you must construct your RECOVER statements according to the following rules:

- When you specify redirection of the recovery output in a RECOVER TABLESPACE statement, you append the keyword INDEPENDENT (which you can abbreviate to INDEP) followed by OUTSPACE and (optionally) the model data set name (dataSetName) to the statement as follows:

```sql
RECOVER TABLESPACE tableSpaceSpecification
   INDEP OUTSPACE optional dataSetName model
   other recovery options
```
When you specify redirection of the recovery output in a REBUILD INDEX or RECOVER BUILDINDEX statement, you insert INDEP OUTSPACE before the table space specification, as follows:

```
REBUILD INDEX ... INDEP OUTSPACE optional dataSetName model
  optional tableSpaceSpecification
  other recovery options
RECOVER BUILDINDEX ... INDEP OUTSPACE
  optional dataSetName model
  optional tableSpaceSpecification
  other recovery options
```

When you rebuild indexes from data that resides in an independent data set, you must also redirect the input for the index rebuild as follows:

```
REBUILD INDEX ... INDEP OUTSPACE optional dataSetName model
  optional tableSpaceSpecification
  INDEP INTO TABLESPACE optional dataSetName model
  other recovery options
```

In addition, if the SYSIN statement in which the REBUILD INDEX appears also includes a RECOVER TABLESPACE statement that specifies redirection, any data set name model specified with INDEP INTABLESPACE must agree with the RECOVER TABLESPACE data set name model. If you do not specify INDEP INTABLESPACE explicitly, RECOVER PLUS assumes it and defaults to the RECOVER TABLESPACE data set name model for index data input.

RECOVER PLUS does not allow

— the recovery of indexes into DB2 data sets when the corresponding table space recovery is to a non-DB2 (independent) data set

— the recovery of indexes into DB2 data sets when the source data for the table space is a non-DB2 data set

That is, if you specify INDEP INTABLESPACE in a REBUILD INDEX statement, you must also specify INDEP OUTSPACE. Similarly, if an index rebuild implies an INDEP INTABLESPACE (because it is in the same SYSIN statement as a RECOVER TABLESPACE with an INDEP OUTSPACE clause for the related table space), the REBUILD INDEX must also have an INDEP OUTSPACE clause.
If you are extracting index keys (using a RECOVER UNLOADKEYS statement) from data that has already been redirected to an independent data set, you insert INDEP INTABLESPACE after the table space specification as follows:

```
RECOVER UNLOADKEYS ... optional table space specification
INDEP INTABLESPACE optional dataSetName model
other recovery options
```
Example 10: Overriding installation options

This example demonstrates the use of the OPTIONS statement to override installation options at execution time. Not all installation options can be overridden. See “OPTIONS” on page 109 for a description of the OPTIONS statement.

NOTE

Some OPTIONS keywords are different from their corresponding installation option names. For example, the OPTIONS keyword is MAXDRIVES, but the installation option is MAXDRIVE. Be sure to use the keywords as they are shown in the OPTIONS syntax diagram in “OPTIONS” on page 109.

Example 10A: Overriding several installation options

In the first part of this example (Figure 76), the SORTDEVT and SORTNUM options are overridden to SYSDA and 03. In addition, the resources eligible for use in this recovery are limited to the archive log 1 and active log 1, and the remote primary and local primary image copies. Note that, in the example output, only an active log was read because it had not yet been archived and was therefore the only source of log records for the recovery.

Figure 76   Example 10A—JCL for overriding installation options

```plaintext
//AFREX12A JOB (PAFR), 'EXAMPLE 10A',
//         CLASS=Q, NOTIFY=&SYSUID,
//         MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN, REGION=0M,
//             PARM='DGE,EXAMPLE10A,NEW'
//STEPLIB DD DISP=SHR, DSN=product.libraries
//         DD DISP=SHR, DSN=DB2.DSNEXIT
//        DD DISP=SHR, DSN=DB2.DSNLOAD
//SYSIN    DD *
OPTIONS
    SORTDEVT SYSDA SORTNUM 03
    RESOURCE SELECTION
    LOGS (ARC1, ACT1)
    COPIES (RP, LP)
RECOVER TABLESPACE BMCDBSMP.BMCTS012
/*
//
```
Example 10B: Overriding the IXRECP installation option

The following JCL (Figure 77) shows the use of the OPTIONS statement to override the IXRECP installation option. IXRECP YES is used to set RECP or RBDP for the indexes associated with the recovered table space to force index recoveries before the data can be accessed. RECOVER PLUS also issues a warning message for each index that has not been recovered with the table space.

Figure 77  Example 10B—JCL for overriding the IXRECP installation option

```
//AFREX12B JOB (PAFR),'EXAMPLE 10B',
//        CLASS=Q,NOTIFY=&SYSUID,
//        MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN,REGION=0M,
//             PARM='DGE,EXAMPLE10B,NEW/RESTART,MSGLEVEL(2)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//        DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN     DD *
OPTIONS IXRECP YES
RECOVER TABLESPACE BMCDBSMP.BMCTS012
/TOCOPY LASTCOPY
/*
```

Figure 78 through Figure 81 on page 382 show the following output, which is produced by running this JCL:

- AFRSTMT
- AFRPRINT
- AFRPLAN
- AFRERR

See Table 10 on page 277 for a description of each of these files.

Figure 78  Example 10B—Edited AFRPRINT for overriding the IXRECP installation option (part 1 of 4)
### Example 10B: Overriding the IXRECP installation option

**Figure 78  Example 10B—Edited AFRPRINT for overriding the IXRECP installation option (part 2 of 4)**

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC405931</td>
<td>START MERGE COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC401021</td>
<td>MERGE TABLESPACE STEP STARTING</td>
</tr>
<tr>
<td>BMC407951</td>
<td>FULL COPY DATASET IS MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00</td>
</tr>
<tr>
<td>BMC407991</td>
<td>MERGE STARTING FOR DATA SET DSNDXW.DSNDBD.AFREX01.TSEX1P1.10001.A001</td>
</tr>
<tr>
<td>BMC405931</td>
<td>INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC405931</td>
<td>SPIN COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409021</td>
<td>TOTAL WAITS = 5, TOTAL WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409811</td>
<td>FULL IC MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00 READ WAITS = 4, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409051</td>
<td>SPACE WRITE WAITS = 1, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409851</td>
<td>NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 27, USED = 27</td>
</tr>
<tr>
<td>BMC407991</td>
<td>MERGE STARTING FOR DATA SET DSNDXW.DSNDBD.AFREX01.TSEX1P1.10001.A001</td>
</tr>
<tr>
<td>BMC409021</td>
<td>INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409021</td>
<td>SPIN COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409021</td>
<td>TOTAL WAITS = 5, TOTAL WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409811</td>
<td>FULL IC MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00 READ WAITS = 4, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409051</td>
<td>SPACE WRITE WAITS = 1, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409851</td>
<td>NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 27, USED = 27</td>
</tr>
<tr>
<td>BMC407981</td>
<td>NUMBER OF PAGES OUTPUT IS = 27</td>
</tr>
<tr>
<td>BMC407951</td>
<td>MML CPOINT TAKEN FOR MERGE PHASE</td>
</tr>
<tr>
<td>BMC401021</td>
<td>MERGE PHASE COMPLETE. ELAPSED TIME = 00:00:01, TIME SINCE UTILITY START = 00:00:07</td>
</tr>
<tr>
<td>BMC408681</td>
<td>MERGE PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC403521</td>
<td>DDNAME SYS00084 (DSN DSNDXW.DSNDBD.AFREX01.TSEX1P1.I0001.A001) WAS SUCCESSFULLY UNALLOCATED</td>
</tr>
<tr>
<td>BMC403561</td>
<td>AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.TSEX1P1.10001.A002</td>
</tr>
<tr>
<td>BMC403561</td>
<td>AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.TSEX1P1.10001.A002</td>
</tr>
<tr>
<td>BMC403501</td>
<td>DATA SET DSNDXW.DSNDBC.AFREX01.TSEX1P1.10001.A002 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00084 ELAPSED = 00:00:00</td>
</tr>
</tbody>
</table>

---

**Figure 78  Example 10B—Edited AFRPRINT for overriding the IXRECP installation option (part 3 of 4)**

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC405931</td>
<td>START MERGE COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC401021</td>
<td>MERGE TABLESPACE STEP STARTING</td>
</tr>
<tr>
<td>BMC407951</td>
<td>FULL COPY DATASET IS MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00</td>
</tr>
<tr>
<td>BMC407991</td>
<td>MERGE STARTING FOR DATA SET DSNDXW.DSNDBD.AFREX01.TSEX1P1.10001.A001</td>
</tr>
<tr>
<td>BMC405931</td>
<td>INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC405931</td>
<td>SPIN COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409021</td>
<td>TOTAL WAITS = 5, TOTAL WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409811</td>
<td>FULL IC MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00 READ WAITS = 4, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409051</td>
<td>SPACE WRITE WAITS = 1, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409851</td>
<td>NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 27, USED = 27</td>
</tr>
<tr>
<td>BMC407981</td>
<td>NUMBER OF PAGES OUTPUT IS = 27</td>
</tr>
<tr>
<td>BMC407951</td>
<td>MML CPOINT TAKEN FOR MERGE PHASE</td>
</tr>
<tr>
<td>BMC401021</td>
<td>MERGE PHASE COMPLETE. ELAPSED TIME = 00:00:01, TIME SINCE UTILITY START = 00:00:07</td>
</tr>
<tr>
<td>BMC408681</td>
<td>MERGE PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC403521</td>
<td>DDNAME SYS00084 (DSN DSNDXW.DSNDBD.AFREX01.TSEX1P1.I0001.A001) WAS SUCCESSFULLY UNALLOCATED</td>
</tr>
<tr>
<td>BMC403561</td>
<td>AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.TSEX1P1.10001.A002</td>
</tr>
<tr>
<td>BMC403561</td>
<td>AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.TSEX1P1.10001.A002</td>
</tr>
<tr>
<td>BMC403501</td>
<td>DATA SET DSNDXW.DSNDBC.AFREX01.TSEX1P1.10001.A002 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00084 ELAPSED = 00:00:00</td>
</tr>
</tbody>
</table>

---

**Figure 78  Example 10B—Edited AFRPRINT for overriding the IXRECP installation option (part 4 of 4)**

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC405931</td>
<td>START MERGE COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC401021</td>
<td>MERGE TABLESPACE STEP STARTING</td>
</tr>
<tr>
<td>BMC407951</td>
<td>FULL COPY DATASET IS MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00</td>
</tr>
<tr>
<td>BMC407991</td>
<td>MERGE STARTING FOR DATA SET DSNDXW.DSNDBD.AFREX01.TSEX1P1.10001.A001</td>
</tr>
<tr>
<td>BMC405931</td>
<td>INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC405931</td>
<td>SPIN COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409021</td>
<td>TOTAL WAITS = 5, TOTAL WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409811</td>
<td>FULL IC MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00 READ WAITS = 4, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409051</td>
<td>SPACE WRITE WAITS = 1, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC409851</td>
<td>NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 27, USED = 27</td>
</tr>
<tr>
<td>BMC407981</td>
<td>NUMBER OF PAGES OUTPUT IS = 27</td>
</tr>
<tr>
<td>BMC407951</td>
<td>MML CPOINT TAKEN FOR MERGE PHASE</td>
</tr>
<tr>
<td>BMC401021</td>
<td>MERGE PHASE COMPLETE. ELAPSED TIME = 00:00:01, TIME SINCE UTILITY START = 00:00:07</td>
</tr>
<tr>
<td>BMC408681</td>
<td>MERGE PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC403521</td>
<td>DDNAME SYS00084 (DSN DSNDXW.DSNDBC.AFREX01.TSEX1P1.10001.A002) WAS SUCCESSFULLY UNALLOCATED</td>
</tr>
<tr>
<td>BMC403561</td>
<td>AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.TSEX1P1.10001.A002</td>
</tr>
<tr>
<td>BMC403561</td>
<td>AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.TSEX1P1.10001.A002</td>
</tr>
<tr>
<td>BMC403501</td>
<td>DATA SET DSNDXW.DSNDBC.AFREX01.TSEX1P1.10001.A002 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00084 ELAPSED = 00:00:00</td>
</tr>
</tbody>
</table>

---

(RECOVER PLUS for DB2 Reference Manual)
Example 10B: Overriding the IXRECP installation option

Figure 78  Example 10B—Edited AFRPRINT for overriding the IXRECP installation option (part 3 of 4)

<table>
<thead>
<tr>
<th>BMC40060I</th>
<th>PENDING FLAG RBDP SET FOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>INDEX = AFR.ICEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40898W</td>
<td>INDEX LEFT IN RBDP BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT</td>
</tr>
<tr>
<td>BMC40060I</td>
<td>PENDING FLAG RBDP* SET FOR</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>INDEX = AFR.INEX1P1</td>
</tr>
<tr>
<td>BMC40898W</td>
<td>INDEX LEFT IN RBDP* BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT</td>
</tr>
<tr>
<td>BMC40012I</td>
<td>MERGE PHASE COMPLETE, ELAPSED TIME = 00:00:00, TIME SINCE UTILITY START = 00:00:08</td>
</tr>
<tr>
<td>BMC40868I</td>
<td>MERGE PHASE COMPLETE, ACCUMULATED TCB TIME = 00:00:00</td>
</tr>
</tbody>
</table>

BMC40016I MERGE PHASE STARTING 01/04/2011 15:32:39
BMC40301I TABLESPACE = AFREX01.TSEX1P1
BMC40305I DSNUM = 3
BMC40484I MESSAGES FROM PHASE SUBTASK:
BMC40417I 1 VSAM DATA SETS NOW ALLOCATED
BMC40060I PENDING FLAG RECP SET FOR
BMC40301I SPACE = AFREX01.TSEX1P1
BMC40305I DSNUM = 3
BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.TSEX1P1.I0001.A003
BMC40356I AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.AFREX01.TSEX1P1.I0001.A003
BMC40350I DATA SET DSNDXW.DSNDBD.AFREX01.TSEX1P1.I0001.A003 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00087 ELAPSED = 00:00:00
BMC40485I END OF PHASE SUBTASK MESSAGES
BMC406304I THIS PHASE WILL RUN IN EXECUTION SUBTASK 1
BMC40593I START MERGE COMPLETE, ELAPSED TIME = 00:00:00
BMC40102I MERGE TABLESPACE STEP STARTING
BMC40795I FULL COPY DATASET IS MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00
BMC40799I MERGE STARTING FOR DATA SET DSNDXW.DSNDBD.AFREX01.TSEX1P1.I0001.A003
BMC40593I INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00
BMC40593I SPIN COMPLETE, ELAPSED TIME = 00:00:00
BMC40902I TOTAL WAITS = 1, TOTAL WAIT TIME = 00:00:00
BMC40981I FULL IC MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00 READ WAITS = 4, WAIT TIME = 00:00:00
BMC40905I SPACE WRITE WAITS = 1, WAIT TIME = 00:00:00
BMC40985I NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 27, USED = 27
BMC40790I NUMBER OF PAGES READ FROM FULL COPY TOTAL = 81, USED = 81
BMC40798I NUMBER OF PAGES OUTPUT IS = 27
BMC40901I MERGE TABLESPACE STEP COMPLETE
BMC40593I PROCESS COMPLETE, ELAPSED TIME = 00:00:00
BMC40352I DDNAME SY500087 (DSN DSNDXW.DSNDBD.AFREX01.TSEX1P1.I0001.A003) WAS SUCCESSFULLY UNALLOCATED
BMC40060I PENDING FLAG RECP RESET FOR
BMC40345I SPACE = AFREX01.TSEX1P1
BMC40305I DSNUM = 3
BMC40060I PENDING FLAG RBDP SET FOR
BMC40301I INDEX = AFR.IEX1P1
BMC40305I DSNUM = 3
BMC40898W INDEX LEFT IN RBDP BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |
BMC40060I PENDING FLAG RBDP* SET FOR |
BMC40301I INDEX = AFR.INEX1P1 |
BMC40898W INDEX LEFT IN RBDP* BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |
BMC40091I 01/04/2011 15:32:39 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE |
BMC40012I MERGE PHASE COMPLETE, ELAPSED TIME = 00:00:00, TIME SINCE UTILITY START = 00:00:08 |
BMC40868I MERGE PHASE COMPLETE, ACCUMULATED TCB TIME = 00:00:00 |

BMC40016I MERGE PHASE STARTING 01/04/2011 15:32:39 |
BMC40301I TABLESPACE = AFREX01.TSEX1P1 |
BMC40305I DSNUM = 4 |
| | | |
Figure 78  Example 10B—Edited AFRPRINT for overriding the IXRECP installation option
(part 4 of 4)

| BMC40016I | MERGE PHASE STARTING 01/04/2011 15:32:49 |
| BMC40301I | TABLESPACE = AFRX01.TSEX1P1 |
| BMC40305I | DSNUM = 24 |

BMC40484I MESSAGES FROM PHASE SUBTASK:

BMC40060I PENDING FLAG RECP SET FOR
BMC40345I SPACE = AFRX01.TSEX1P1
BMC40305I DSNUM = 24
BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBD.AFRX01.TSEX1P1.IO001.A024
BMC40356I AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBD.AFRX01.TSEX1P1.IO001.A024
BMC40350I DATA SET DSNDXW.DSNDBD.AFRX01.TSEX1P1.IO001.A024 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00129 ELAPSED = 00:00:00
BMC40417I 2 VSAM DATA SETS NOW ALLOCATED

BMC40485I END OF PHASE SUBTASK MESSAGES

BMC40593I INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00

BMC40060I PENDING FLAG RECP SET FOR
BMC40345I SPACE = AFRX01.TSEX1P1
BMC40305I DSNUM = 24
BMC40060I PENDING FLAG RBDP SET FOR
BMC40303I INDEX = AFR.ICSEX1P1
BMC40305I DSNUM = 24
BMC40303I INDEX = AFR.INEX1P1
BMC40305I DSNUM = 24
BMC40091I 01/04/2011 15:32:49 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE

BMC40020I BYT:0, 0, 0; BLK:0, 0, 0; TOT:430968, 430968, 430968; iter: 1; RC:4

Figure 79  Example 10B—AFRSTMT for overriding the IXRECP installation option
(part 1 of 4)

BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC36173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40937I z/OS RELEASE = 1.9.00, PID = HBB7740, SMS RELEASE = 1.9.0
BMC40157I AVAILABLE REGION BELOW 16M = 8836K, AVAILABLE REGION ABOVE 16M = 1574392K, NUMBER OF CPUS = 3
BMC40020I BYT:0, 0, 0; BLK:0, 0, 0; TOT:430968, 430968, 430968; iter: 1; RC:4
BMC40931I PARM LIST: DB2 SSID = DXW
BMC40930I PARM NOT CODED - UTILID
BMC40009I UTILITY NEW/RESTART REQUESTED - SYNCPOINT RESTART WILL OCCUR IF UTILID IS IN BMCUTIL
BMC40933I RESTART = NEW/RESTART
BMC40934I MSGLEVEL = 2
Example 10B: Overriding the IXRECP installation option

Example 10B—AFRSTMT for overriding the IXRECP installation option  (part 2 of 4)

| BMC40930I | PARM NOT CODED - CHECKPT |
| BMC40930I | PARM NOT CODED - RD2STAT |
| BMC40930I | PARM NOT CODED - AFROPTS |
| BMC40930I | PARM NOT CODED - ACAOPTS |
| BMC40924I | OPTIONS: PLANRECV = AFRB101T (MAIN PLAN) |
| BMC40925I | TBUFFS = 1000 (BUFFER MANAGER PAGES) |
| BMC40944I | OPRBDZID = YES (ACQUIRE RACF AUTHORITY OF DB2) |
| BMC40926I | CHECKPT = PHASE (CHECKPOINTING) |
| BMC40945I | CHECKINT = 0 (CHECKPOINT INTERVAL MINUTES) |
| BMC40927I | RD2STAT = YES (RESET DB2 OBJECT STATUS) |
| BMC40946I | AMSCAT = NO (USE 'CATALOG' IN IDCAMS INPUT) |
| BMC40928I | WKUNIT = SYSALLDA (WORK UNIT NAME) |
| BMC40956I | RCLTSK = 10 (MAXIMUM RECALL TASKS) |
| BMC40941I | IXRECP = NO (SET INDEX RECOVER PENDING) |
| BMC40911I | MAXDRIVES = 0 (DEFAULT MAX TAPE DRIVES) |
| BMC40921I | MAXLOGS = 3 (DEFAULT MAX CONCURRENT LOG FILES) |
| BMC40942I | MAXKSORT = 6 (DEFAULT) (MAX CONCURRENT INDEX SORTS) |
| BMC96230I | KSORTSHARE = YES (KEY SORTS SHARED BY ALL TASKS) |
| BMC96192I | MAXLSORT = 50 (MAX CONCURRENT LOG SORTS) |
| BMC40962I | SORTDEVT = SORT DEFAULT (DEVICE TYPE FOR LOG AND INDEX SORTS) |
| BMC40963I | SORTNUM = SORT DEFAULT (WORK DATASETS FOR LOG AND INDEX SORTS) |
| BMC40964I | OUTCOPY = ASCODED (OUTCOPY CREATION DIRECTIVE) |
| BMC40547I | INDEXLOG = NO (DEFAULT USE OF INDEXLOG STRATEGY) |
| BMC40878I | WTOR = YES (ISSUE WTOR IF SPACE REMAINS IN STOPP STATUS) |
| BMC40537I | ERRCONT = 10 (DEFAULT MAX SEVERE ERRORS) |
| BMC96221I | AUTOSIZE = YES (AUTO SIZE ACCUM OUTPUT FILES) |
| BMC40481I | PUBLICPLAN = YES (GRANT PLAN TO PUBLIC) |
| BMC40488I | BINDQUALIFIER = BMCAF (QUALIFIER USED FOR DYNAMIC BIND) |
| BMC96233I | BACKOUT = AUTO (DEFAULT POINT-IN-TIME RECOVERY STRATEGY) |
| BMC96235I | AUX = NO (INCLUDE XML AND/OR LOB AUXILIARY OBJECTS) |
| BMC40967I | TAPE DEVICE01 = 3590-1 |
| BMC40967I | TAPE DEVICE02 = 3490 |
| BMC40967I | TAPE DEVICE03 = 3480X |
| BMC40967I | TAPE DEVICE04 = ATLVS |
| BMC40967I | TAPE DEVICE05 = ATLS902 |
| BMC40967I | TAPE DEVICE06 = CART |
| BMC40967I | TAPE DEVICE07 = CARTVS |
| BMC40967I | TAPE DEVICE08 = CRT8NS |
| BMC40967I | TAPE DEVICE09 = CRT9NS |
| BMC40967I | TAPE DEVICE10 = DANBARR |
| BMC40967I | TAPE DEVICE11 = EXT359E |
| BMC40967I | TAPE DEVICE12 = IBMATL |
| BMC40967I | TAPE DEVICE13 = IBMVTS |
| BMC40967I | TAPE DEVICE14 = SA3590 |
| BMC40967I | TAPE DEVICE15 = TAPABL |
| BMC40967I | TAPE DEVICE16 = 3590B |
| BMC40967I | TAPE DEVICE17 = 3590E |
| BMC40967I | TAPE DEVICE18 = 3480 |
| BMC40967I | TAPE DEVICE19 = 3400-9 |
| BMC40967I | TAPE DEVICE20 = SYS3480R |
| BMC40967I | TAPE DEVICE21 = SYS348XR |
| BMC40929I | SORT VALUES: SORTCORE = 0 K (SORT MEMORY VALUE) |
| BMC40947I | VSCORSET = 0 K (TOTAL MEMORY OVERRIDE) |
| BMC96111I | DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE |
| BMC96131I | COPIES = (LP,LB) (DEFAULT SEQUENCE FOR IMAGE COPIES) |
| BMC96141I | ACCUMS = (LP,LB) (DEFAULT SEQUENCE FOR CA FILES) |
| BMC96121I | DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE |
| BMC96131I | COPIES = (RP,RB) (DEFAULT SEQUENCE FOR IMAGE COPIES) |
| BMC96141I | ACCUMS = (RP,RB) (DEFAULT SEQUENCE FOR CA FILES) |
| BMC40018I | RECOVER PLUS FOR DB2 V10.1.00 |
| BMC40876I | MAINT: NO RECOVER PLUS PTFS APPLIED |
| BMC40910I | SOLUTION COMMON CODE V10.1.00 |
| BMC40876I | MAINT: NO SOLUTION COMMON CODE PTFS APPLIED |
| BMC96010I | BMCSORT ENGINE V2.3.01 |
Example 10B: Overriding the IXRECP installation option

Figure 79 Example 10B—AFRSTMT for overriding the IXRECP installation option (part 3 of 4)

```
BMC40876I MAINT: BPJ0279 BPJ0288
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276

BMC40335I PACKAGE CHECKING AND BINDING STARTING 01/04/2011 15:32:31
BMC40336I PACKAGE CHECKING AND BINDING FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40954I AVAILABLE MEMORY BELOW THE LINE = 8744K, TOTAL AVAILABLE MEMORY (AT LEAST) = 1582700K
BMC40020I BYT:0, 0, 0; BLK:0, 0, 0; TOT:430968, 430968, 430968;iter: 1; RC:4
BMC40037I UTILID MVSMAR1.AFR10B WAS NOT FOUND IN THE BMCUTIL TABLE - UTILITY WILL RUN AS IF 'NEW' WERE SPECIFIED

BMC40335I OPTIONS IXRECP YES
BMC40101I RECOVER TABLESPACE AFREX01.TSEX1P1
BMC40101I TOCOPY LASTCOPY
BMC40101I

BMC40335I CATALOG LOOKUP STARTING 01/04/2011 15:32:31
BMC40336I PROC RO STMT FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40336I SCAN SYSTB FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40336I SCAN SYSEX FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40336I TS LOOKUP FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40336I DB LOOKUP FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40336I TB LOOKUP FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40336I SG LOOKUP FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40336I VOLUMES IN STOGROUP SYSDEFLT WILL BE USED IN THE FOLLOWING ORDER:
BMC40336I       (<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>)
BMC40336I SCAN SYSVOL FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40070I VOLUMES IN STOGROUP SYSDEFLT WILL BE USED IN THE FOLLOWING ORDER:
BMC40070I       (<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>)
BMC40336I SCAN SYSVOL FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40336I CHK DS FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40336I HDR PG RD FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME    = 00:00:00, UTILITY CPU TIME    = 00:00:00

BMC40336I SYSCOPY LOOKUP FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
```
Example 10B: Overriding the IXRECP installation option

Figure 79  Example 10B—AFRSTMT for overriding the IXRECP installation option (part 4 of 4)

| BMC40288I | CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00 |
| BMC40336I | CATALOG LOOKUP FINISHED. |
| BMC40287I | ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01 |
| BMC40288I | CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00 |

BMC40335I SYNCHRONIZATION STARTING 01/04/2011 15:32:31
BMC40025I OUTPUT FROM COMMAND ‘STO DB(AFREX01) SPACE(TSEX1P1) AT (COMMIT)’ FOLLOWS:
BMC40020I DSN9022I *DXW3 DSNTDDIS 'STOP DATABASE' NORMAL COMPLETION
BMC40336I SYNCHRONIZATION FINISHED.
BMC40287I ELAPSED TIME = 00:00:02, UTILITY ELAPSED TIME = 00:00:03
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC96306I MAXLSORT REDUCED FROM 50 TO 7 DUE TO MEMORY CONSTRAINTS
BMC40336I DDSCAN FINISHED.
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:03
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40335I PLANNING STARTING 01/04/2011 15:32:34
BMC40025I OUTPUT FROM COMMAND ‘STA DB(AFREX01) SPACE(TSEX1P1) ACCESS(RW)’ FOLLOWS:
BMC40020I DSN9022I *DXW3 DSNTDDIS 'START DATABASE' NORMAL COMPLETION

Figure 80  Example 10B—Edited AFRPLAN for overriding the IXRECP installation option (part 1 of 4)

BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40700I EXECUTION PLAN SUMMARY:

BMC40701I PHASE:  MERGE
BMC40702I PHASE INITIALIZATION:
BMC40301I TABLESPACE = AFREX01.TSEX1P1
BMC40305I DSNUM = 1
BMC40738I ALLOCATE INPUT IMAGE COPY FILE
BMC40302I DSN = MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00
BMC40738I FILE OPTIMIZATION
BMC40738I ALLOCATE 2 INPUT FILE BUFFER(S)
BMC40738I OPEN IMAGE COPY FILE
BMC40738I DSN = MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00
BMC40738I ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I ALLOCATE SPACEMAP BUFFER (4096 BYTES)
BMC40738I ALLOCATE PASTE BUFFER (4096 BYTES)
BMC40738I SET RECOVERY PENDING (RECP) FLAG
BMC40738I CREATE SPACE
BMC40345I SPACE = AFREX01.TSEX1P1
BMC40305I DSNUM = 1
BMC40738I ALLOCATE SPACE (READ/WRITE)
BMC40345I SPACE = AFREX01.TSEX1P1
BMC40305I DSNUM = 1
BMC40738I CONNECT TO SPACE
BMC40738I ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I FILE OPTIMIZATION
BMC40738I ALLOCATE 2 SPACE OUTPUT BUFFER(S)
BMC40738I ALLOCATE SPACEMAP BUFFER (4096 BYTES)
BMC40738I ALLOCATE PASTE BUFFER (4096 BYTES)
BMC40702I STEP:  MERGE SPACE
BMC40738I MERGE SPACE
Figure 80  Example 10B—Edited AFRPLAN for overriding the IXRECP installation option (part 2 of 4)

<table>
<thead>
<tr>
<th>BMC40301I</th>
<th>TABLESPACE = AFREX01.TSEX1P1</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>REGISTER PARTIAL RECOVERY IN SYSCOPY</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET INDEX(S) STATUS TO REBUILD PENDING IF NEEDED</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
<tr>
<td>BMC40701I</td>
<td>PHASE: MERGE</td>
</tr>
<tr>
<td>BMC40721I</td>
<td>PHASE INITIALIZATION:</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40702I</td>
<td>STEP: MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>REGISTER PARTIAL RECOVERY IN SYSCOPY</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET INDEX(S) STATUS TO REBUILD PENDING IF NEEDED</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
<tr>
<td>BMC40701I</td>
<td>PHASE: MERGE</td>
</tr>
<tr>
<td>BMC40721I</td>
<td>PHASE INITIALIZATION:</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
</tbody>
</table>
## Example 10B: Overriding the IXRECP installation option

### Chapter 5 Examples of RECOVER PLUS jobs

### Figure 80  Example 10B—Edited AFRPLAN for overriding the IXRECP installation option  (part 3 of 4)

<table>
<thead>
<tr>
<th>BMC40738I</th>
<th>ALLOCATE PASTE BUFFER (4096 BYTES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40702I</td>
<td>STEP: MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACE MAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>REGISTER PARTIAL RECOVERY IN SYSCOPY</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET INDEX(S) STATUS TO REBUILD PENDING IF NEEDED</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40711I</th>
<th>PHASE: MERGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40721I</td>
<td>PHASE INITIALIZATION:</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 4</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 4</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACE MAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40702I</td>
<td>STEP: MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 4</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACE MAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>REGISTER PARTIAL RECOVERY IN SYSCOPY</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET INDEX(S) STATUS TO REBUILD PENDING IF NEEDED</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>

...
Example 10B: Overriding the IXRECP installation option

Figure 80  Example 10B—Edited AFRPLAN for overriding the IXRECP installation option (part 4 of 4)

<table>
<thead>
<tr>
<th>BMC40305I</th>
<th>DSNUM = 24</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40738I</td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>STEP: MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>TABLESPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>SPACE = AFREX01.TSEX1P1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 24</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CLOSE IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 INPUT FILE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE INPUT IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = MVSMAR1.LP.AFREX01.TSEX1P1.F000.G0020V00</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>REGISTER PARTIAL RECOVERY IN SYSCOPY</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET INDEX(S) STATUS TO REBUILD PENDING IF NEEDED</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>

Figure 81  Example 10B—AFRRERR for overriding the IXRECP installation option

<table>
<thead>
<tr>
<th>BMC40018I</th>
<th>RECOVER PLUS FOR DB2 V10.1.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40473I</td>
<td>COPYRIGHT BMC SOFTWARE INC. 1991-2011</td>
</tr>
<tr>
<td>BMC96173I</td>
<td>RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884</td>
</tr>
<tr>
<td>BMC40474I</td>
<td>CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO <a href="mailto:SUPPORT@BMC.COM">SUPPORT@BMC.COM</a></td>
</tr>
</tbody>
</table>

| BMC40898W | INDEX LEFT IN RBDP BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |
| BMC40898W | INDEX LEFT IN RBDP* BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |
| BMC40898W | INDEX LEFT IN RBDP BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |
| BMC40898W | INDEX LEFT IN RBDP* BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |
| BMC40898W | INDEX LEFT IN RBDP BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |
| BMC40898W | INDEX LEFT IN RBDP* BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |
| BMC40898W | INDEX LEFT IN RBDP BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |
| BMC40898W | INDEX LEFT IN RBDP* BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |
| BMC40898W | INDEX LEFT IN RBDP BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |
| BMC40898W | INDEX LEFT IN RBDP* BECAUSE TABLESPACE WAS RECOVERED TO POINT-IN-TIME AND INDEX WAS NOT |

Example 11: Recovering using a change accumulation file

This example illustrates the creation and use of change accumulation files in the recovery of a table space. An index on the table in the table space is also rebuilt.

Example 11A: Creating the change accumulation file

The JCL in Figure 82 creates the change accumulation file.

Figure 82  Example 11A—JCL using R+/CHANGE ACCUM to accumulate log records

```
//AFREX13A JOB (PAFR),'EXAMPLE 11A', 
//         CLASS=Q,NOTIFY=SYSUID, 
//         MSGCLASS=X 
//*
//RECOVER EXEC PGM=AFRMAIN,REGION=OM, 
//   PARM='DEBA,EXAMPLE11A,NEW/RESTART,MSGLEVEL(2)' 
//STEPLIB DD DISP=SHR,DSN=product.libraries 
//   DD DISP=SHR,DSN=DB2.DSNEXIT 
//   DD DISP=SHR,DSN=DB2.DSNLOAD 
//SYSIN DD * 
ACCUM GROUP BMCSMPGP
/*
//
```

Example 11B: Using the change accumulation file

The JCL in Figure 83 recovers the table space using the change accumulation file created in “Example 11A: Creating the change accumulation file.”

Figure 83  Example 11B—JCL recovering a table space using a change accumulation file

```
//AFREX13B JOB (PAFR),'EXAMPLE 11B', 
//         CLASS=Q,NOTIFY=SYSUID, 
//         MSGCLASS=X 
//*
//ROUTE XEQ BMCPLX1 
//*JOBPARM SYSAFF=DB2A 
//*
//RECOVER EXEC PGM=AFRMAIN,REGION=OM, 
//   PARM='DEBA,EXAMPLE11B,NEW/RESTART,MSGLEVEL(2)' 
//STEPLIB DD DISP=SHR,DSN=product.libraries 
//   DD DISP=SHR,DSN=DB2.DSNEXIT 
//   DD DISP=SHR,DSN=DB2.DSNLOAD 
//SYSIN DD *
//OPTIONS USEACCUM YES 
RECOVER TABLESPACE BMCDBSMP.BMCTS020 
REBUILD INDEX (RDASTC.BMCIX020) NOWORKDDN 
/*
//
```
Example 12: Using the MAINT parameter

This example illustrates the use of the MAINT parameter on the EXEC statement to print a record of applied maintenance to AFRSUMRY and AFRSTMT. Common tables used by BMC utilities are also printed to AFRSTMT.

The AFRSUMRY and AFRSTMT output produced by running this JCL follows. See Table 10 on page 277 for a description of each of these files. Notice that you see the maintenance that has been applied for RECOVER PLUS and each component or technology used by RECOVER PLUS.
Chapter 5  Examples of RECOVER PLUS jobs 385

Example 12: Using the MAINT parameter

| BMC049297I | 2/5/5.5.5, PID = H87740, SMS RELEASE = 1.9.0 |
| BMC04925I  | AVAILABLE REGION BELOW 16M = 8836K, AVAILABLE REGION ABOVE 16M = 1574392K, NUMBER OF CPUS = 3 |
| BMC049202I | BYT:0, 0, 0; BLK:0, 0, 0; TOT:491836, 491836, 491836; iter: 1; RC:4 |
| BMC04931I  | PARM LIST: DB2 SSID = DXW |
| BMC04932I  | UTILID = EXAMPLE12 |
| BMC04933I  | RECOVER PLUS MAINTENANCE DISPLAY REQUESTED |
| BMC04934I  | RESTART = MAINT |
| BMC04935I  | MSGLEVEL = 1 |
| BMC04930I  | PARM NOT CODED - CHECKPT |
| BMC04930I  | PARM NOT CODED - RDB2STAT |
| BMC04930I  | PARM NOT CODED - AFROPTS |
| BMC04930I  | PARM NOT CODED - ACAOPTS |
| BMC04924I  | OPTIONS: PLANRECV = AFRB101T (MAIN PLAN) |
| BMC04941I  | TBUFFS = 1000 (BUFFER MANAGER PAGES) |
| BMC04944I  | OPND2BID = YES (ACQUIRE RACF AUTHORITY OF DB2) |
| BMC04926I  | CHECKPT = PHASE (CHECKPOINTING) |
| BMC04945I  | CHECKINT = 0 (CHECKPOINT INTERVAL MINUTES) |
| BMC04927I  | RDB2STAT = YES (RESET DB2 OBJECT STATUS) |
| BMC04946I  | AMSCAT = NO (USE 'CATALOG' IN IDCAMS INPUT) |
| BMC04928I  | SMCORE = (OK, OK) (SORT CORE VALUES) |
| BMC04933I  | RESINV = (OK) (MEMORY BELOW 16MB EXCLUDED FROM USE BY SORT) |
| BMC04938I  | WKUNIT = SYSALLDA (WORK UNIT NAME) |
| BMC04956I  | RCLTSK = 10 (MAXIMUM RECALL TASKS) |
| BMC04941I  | IXRECP = NO (SET INDEX RECOVER PENDING) |
| BMC04981I  | MAXDRIVES = 0 (DEFAULT MAX TAPE DRIVES) |
| BMC04982I  | MAXLOGS = 3 (DEFAULT MAX CONCURRENT LOG FILES) |
| BMC04942I  | MAXKSORT = 6 (DEFAULT) (MAX CONCURRENT INDEX SORTS) |
| BMC04923I  | KSORTSHARE = YES (KEY SORTS SHARED BY ALL TASKS) |
| BMC04921I  | MAXSORT = 50 (MAX CONCURRENT LOG SORTS) |
| BMC04962I  | SHORTDVT = SORT DEFAULT (DEVICE TYPE FOR LOG AND INDEX SORTS) |
| BMC04963I  | SORTNUM = SORT DEFAULT (WORK DATASETS FOR LOG AND INDEX SORTS) |
| BMC04964I  | OUTCOPY = ASCODED (OUTCOPY CREATION DIRECTIVE) |
| BMC04547I  | INDEXLOG = NO (DEFAULT USE OF INDEXLOG STRATEGY) |
| BMC04978I  | WTOR = YES (ISSUE WTOR IF SPACE REMAINS IN STOPP STATUS) |
| BMC04937I  | ERRCONT = 10 (DEFAULT MAX SEVERE ERRORS) |
| BMC04922I  | AUTOSIZE = YES (AUTO SIZE ACCUM OUTPUT FILES) |
| BMC04241I  | PUBLICPLAN = YES (GRANT PLAN TO PUBLIC) |
| BMC04098I  | BINDQUALIFIER = BMCANF (QUALIFIER USED FOR DYNAMIC BIND) |
| BMC04923I  | BACKOUT = AUTO (DEFAULT POINT-IN-TIME RECOVERY STRATEGY) |
| BMC04925I  | AUX = NO (INCLUDE XML AND/OR LOB AUXILIARY OBJECTS) |
| BMC04967I  | TAPE DEVICE01 = 3590-1 |
| BMC04967I  | TAPE DEVICE02 = 3490 |
| BMC04967I  | TAPE DEVICE03 = 3480X |
| BMC04967I  | TAPE DEVICE04 = ATLTS52 |
| BMC04967I  | TAPE DEVICE05 = ATLS592 |
| BMC04967I  | TAPE DEVICE06 = CART |
| BMC04967I  | TAPE DEVICE07 = CARTM2 |
| BMC04967I  | TAPE DEVICE08 = CRT8BNS |
| BMC04967I  | TAPE DEVICE09 = CRT9NS |
| BMC04967I  | TAPE DEVICE10 = DANBARR |
| BMC04967I  | TAPE DEVICE11 = EXT3592 |
| BMC04967I  | TAPE DEVICE12 = IBMATL |
| BMC04967I  | TAPE DEVICE13 = IBMVTS |
| BMC04967I  | TAPE DEVICE14 = SA3590 |
| BMC04967I  | TAPE DEVICE15 = TAPEABL |
| BMC04967I  | TAPE DEVICE16 = 3590B |
| BMC04967I  | TAPE DEVICE17 = 3590E |
| BMC04967I  | TAPE DEVICE18 = 3400-9 |
| BMC04967I  | TAPE DEVICE19 = 3590 |
| BMC04967I  | TAPE DEVICE20 = SY33480R |
| BMC04967I  | TAPE DEVICE21 = SY3348XR |
| BMC049291I | SORT VALUES: SORTCORE = 0 K (SORT MEMORY VALUE) |
| BMC04947I  | VSCORET = 0 K (TOTAL MEMORY OVERRIDE) |
| BMC06111I  | DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE |
| BMC06113I  | COPIES = (LP, LB) (DEFAULT SEQUENCE FOR IMAGE COPIES) |
| BMC06114I  | ACCUMS = (LP, LB) (DEFAULT SEQUENCE FOR CA FILES) |
| BMC06121I  | DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE |
Figure 86  Example 12—AFRSTMT using the MAINT parameter  (part 3 of 3)

| BMC96113I | COPIES = (RP,RB)  (DEFAULT SEQUENCE FOR IMAGE COPIES) |
| BMC96114I | ACCUMS = (RP,RB)  (DEFAULT SEQUENCE FOR CA FILES) |
| BMC40018I | RECOVER PLUS FOR DB2 V10.1.00 |
| BMC40876I | MAINT:  NO RECOVER PLUS PTFS APPLIED |
| BMC96010I | SOLUTION COMMON CODE V10.1.00 |
| BMC40876I | MAINT:  NO SOLUTION COMMON CODE PTFS APPLIED |
| BMC40873I | BMCUTIL:  BMCUTIL.CMN_BMCUTIL |
| BMC40873I | BMCSYNC:  BMCUTIL.CMN_BMCSYNC |
| BMC40873I | BMXCOPY:  BMCUTIL.CMN_BMXCOPY |
| BMC40873I | BMCHIST:  BMCUTIL.CMN_BMCHIST |
| BMC40873I | SYSCOPY:  BMCUTIL.CMN_BMSCOPY |
| BMC40873I | SYSLGRNX:  BMCUTIL.CMN_BMCSLGRNX |
| BMC40873I | CADEF:  AFR320D.ACA320D_CADEF |
| BMC40873I | CAFILE:  AFR320D.ACA320D_CAFILE |
| BMC40873I | CAFILECP:  AFR320D.ACA320D_CAFILECP |
| BMC40873I | CAGROUP:  AFR320D.ACA320D_CAGROUP |
| BMC40873I | CAGRPCP:  AFR320D.ACA320D_CAGRCP |
| BMC40873I | CALGRNG:  AFR320D.ACA320D_CALGRNG |
| BMC40873I | CAREPOS:  AFR320D.ACA320D_CAREPOS |

Figure 86  Example 12—AFRSTMT using the MAINT parameter  (part 3 of 3)

| BMC96010I | BMCSORT ENGINE V2.3.01 |
| BMC40876I | MAINT:  BPJ0195  BPJ0198  BPJ0209  BPJ0250  BPJ0263  BPJ0269  BPJ0276 |
| BMC40876I | MAINT:  BPJ0279  BPJ0288 |
| BMC40873I | BMCUTIL:  BMCUTIL.CMN_BMCUTIL |
| BMC40873I | BMCSYNC:  BMCUTIL.CMN_BMCSYNC |
| BMC40873I | BMXCOPY:  BMCUTIL.CMN_BMXCOPY |
| BMC40873I | BMCHIST:  BMCUTIL.CMN_BMCHIST |
| BMC40873I | SYSCOPY:  BMCUTIL.CMN_BMSCOPY |
| BMC40873I | SYSLGRNX:  BMCUTIL.CMN_BMCSLGRNX |
| BMC40873I | CADEF:  AFR320D.ACA320D_CADEF |
| BMC40873I | CAFILE:  AFR320D.ACA320D_CAFILE |
| BMC40873I | CAFILECP:  AFR320D.ACA320D_CAFILECP |
| BMC40873I | CAGROUP:  AFR320D.ACA320D_CAGROUP |
| BMC40873I | CAGRPCP:  AFR320D.ACA320D_CAGRCP |
| BMC40873I | CALGRNG:  AFR320D.ACA320D_CALGRNG |
| BMC40873I | CAREPOS:  AFR320D.ACA320D_CAREPOS |
| BMC40873I | BMCSORT ENGINE V2.3.01 |
| BMC40873I | BMCUTIL:  BMCUTIL.CMN_BMCUTIL |
| BMC40873I | BMCSYNC:  BMCUTIL.CMN_BMCSYNC |
| BMC40873I | BMXCOPY:  BMCUTIL.CMN_BMXCOPY |
| BMC40873I | BMCHIST:  BMCUTIL.CMN_BMCHIST |
| BMC40873I | SYSCOPY:  BMCUTIL.CMN_BMSCOPY |
| BMC40873I | SYSLGRNX:  BMCUTIL.CMN_BMCSLGRNX |
| BMC40873I | CADEF:  AFR320D.ACA320D_CADEF |
| BMC40873I | CAFILE:  AFR320D.ACA320D_CAFILE |
| BMC40873I | CAFILECP:  AFR320D.ACA320D_CAFILECP |
| BMC40873I | CAGROUP:  AFR320D.ACA320D_CAGROUP |
| BMC40873I | CAGRPCP:  AFR320D.ACA320D_CAGRCP |
| BMC40873I | CALGRNG:  AFR320D.ACA320D_CALGRNG |
| BMC40873I | CAREPOS:  AFR320D.ACA320D_CAREPOS |
| BMC40873I | BMCSORT ENGINE V2.3.01 |
| BMC40873I | BMCUTIL:  BMCUTIL.CMN_BMCUTIL |
| BMC40873I | BMCSYNC:  BMCUTIL.CMN_BMCSYNC |
| BMC40873I | BMXCOPY:  BMCUTIL.CMN_BMXCOPY |
| BMC40873I | BMCHIST:  BMCUTIL.CMN_BMCHIST |
| BMC40873I | SYSCOPY:  BMCUTIL.CMN_BMSCOPY |
| BMC40873I | SYSLGRNX:  BMCUTIL.CMN_BMCSLGRNX |
| BMC40873I | CADEF:  AFR320D.ACA320D_CADEF |
| BMC40873I | CAFILE:  AFR320D.ACA320D_CAFILE |
| BMC40873I | CAFILECP:  AFR320D.ACA320D_CAFILECP |
| BMC40873I | CAGROUP:  AFR320D.ACA320D_CAGROUP |
| BMC40873I | CAGRPCP:  AFR320D.ACA320D_CAGRCP |
| BMC40873I | CALGRNG:  AFR320D.ACA320D_CALGRNG |
| BMC40873I | CAREPOS:  AFR320D.ACA320D_CAREPOS |

BMC40954I  AVAILABLE MEMORY BELOW THE LINE = 8744K, TOTAL AVAILABLE MEMORY (AT LEAST) = 1582696K
Example 13: Using the LOGSCAN command

This example (Figure 87) illustrates the use of the LOGSCAN command. LOGSCAN gathers information about log records that you can use to determine your recovery strategy. The LOGSCAN command uses the SYSCAN DD statement included in this example to designate output for the report that the LOGSCAN command generates.

Figure 87 Example 13—JCL using the LOGSCAN command

```bash
//AFREX15 JOB (PAPR), 'EXAMPLE 13',
//  CLASS=Q, NOTIFY=&SYSUID,
//  MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN,REGION=OM,
//  PARM='DGE,EXAMPLE13,NEW/RESTART,MSGLEVEL(2)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN DD *
OPTIONS INDEXLOG YES
LOGSCAN TABLESPACE BMCDBSMP.BMCTS013
INDEXSPACE BMCDBSMP.BMCIX013
TOLOGPOINT LASTQUIESCE BACKOUT
/*
//SYSSCAN DD SYSOUT=* 
```

The SYSSCAN output produced by running this JCL follows (Figure 88).

Figure 88 Example 13—SYSSCAN for the LOGSCAN command (part 1 of 3)
Figure 88  Example 13—SYSSCAN for the LOGSCAN command (part 2 of 3)

| BMC40725I | 5 WAITS FOR THREAD     WAIT TIME = 00:00:00 |
| BMC40172I | 240000 LOG RECORDS AND 24048000 BYTES OF LOG DATA PROCESSED |
| BMC40302I | DSN = DSNDXW.DXW3.LOGCOPY1.DS03.DATA |
| BMC40173I | TIME SPENT IN DATASET ALLOCATION = 00:00:00 |
| BMC40174I | 55 I/O WAITS I/O WAIT TIME = 00:00:00 |
| BMC40172I | 80287 LOG RECORDS AND 8044769 BYTES OF LOG DATA PROCESSED |
| BMC40302I | DSN = DSNDXW.DXW4.LOGCOPY1.DS01.DATA |
| BMC40173I | TIME SPENT IN DATASET ALLOCATION = 00:00:00 |
| BMC40174I | 68 I/O WAITS I/O WAIT TIME = 00:00:00 |
| BMC40172I | 120000 LOG RECORDS AND 12024000 BYTES OF LOG DATA PROCESSED |
| BMC40302I | DSN = DSNDXW.DXW3.LOGCOPY1.DS01.DATA |
| BMC40173I | TIME SPENT IN DATASET ALLOCATION = 00:00:00 |
| BMC40174I | 27 I/O WAITS I/O WAIT TIME = 00:00:00 |
| BMC40172I | 39713 LOG RECORDS AND 3979231 BYTES OF LOG DATA PROCESSED |

| BMC40725I | PROCESSING LOG RANGE: |
| BMC40302I | DSN = DSNDXW.DXW3.ARCLG1.A0017436 |
| BMC40555I | FROMRBA = X'008E857B1000'  TORBA = X'008E85BBCFFF' |
| BMC40724I | DATA SHARING MEMBERID IS 2 |
| BMC40725I | PROCESSING LOG RANGE: |
| BMC40302I | DSN = DSNDXW.DXW3.ARCLG1.A0017437 |
| BMC40555I | FROMRBA = X'008E85BBD000'  TORBA = X'008E8696AD70' |
| BMC40724I | DATA SHARING MEMBERID IS 2 |
| BMC40193I | 3 LOG DATASET BUFFERS OF 737280 BYTES EACH WERE ALLOCATED |
| BMC40302I | DSN = DSNDXW.DXW3.ARCLG1.A0017436 |
| BMC40193I | DSN = DSNDXW.DXW3.ARCLG1.A0017437 |

| BMC40117I | ********** PERFORMANCE STATISTICS FROM LOG READ STEP ********** |
| BMC40120I | 2 FILES READ     80 I/O WAITS I/O WAIT TIME = 00:00:00 |
| BMC40175I | 7 WAITS FOR THREAD     WAIT TIME = 00:00:00 |
| BMC40172I | 124586 LOG RECORDS AND 16029419 BYTES OF LOG DATA PROCESSED |

| BMC40302I | DSN = DSNDXW.DXW3.ARCLG1.A0017436 |
| BMC40173I | TIME SPENT IN DATASET ALLOCATION = 00:00:00 |
| BMC40174I | 80 I/O WAITS I/O WAIT TIME = 00:00:00 |
| BMC40172I | 27521 LOG RECORDS AND 3642835 BYTES OF LOG DATA PROCESSED |
| BMC40302I | DSN = DSNDXW.DXW3.ARCLG1.A0017437 |
| BMC40173I | TIME SPENT IN DATASET ALLOCATION = 00:00:00 |
| BMC40174I | 0 I/O WAITS I/O WAIT TIME = 00:00:00 |
| BMC40172I | 97065 LOG RECORDS AND 12386584 BYTES OF LOG DATA PROCESSED |

| BMC40850I | OBJECT | TYPE | PART | #RECS | #BYTES |
| BMC40851I | AFREX01.TSEX1P1 | TS | 001 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 002 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 003 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 004 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 005 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 006 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 007 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 008 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 009 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 010 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 011 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 012 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 013 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 014 | 3025 | 220200 |
| BMC40851I | AFREX01.TSEX1P1 | TS | 015 | 3025 | 220200 |
### Example 13: Using the LOGSCAN command

**Figure 88  Example 13—SYSSCAN for the LOGSCAN command (part 3 of 3)**

<table>
<thead>
<tr>
<th>Sysid</th>
<th>Logname</th>
<th>Type</th>
<th>Recid</th>
<th>Size</th>
<th>Start</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40851I</td>
<td>AFREX01.TSEX1P1</td>
<td>TS</td>
<td>016</td>
<td>3025</td>
<td>220200</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.TSEX1P1</td>
<td>TS</td>
<td>017</td>
<td>3025</td>
<td>220200</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.TSEX1P1</td>
<td>TS</td>
<td>018</td>
<td>3025</td>
<td>220200</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.TSEX1P1</td>
<td>TS</td>
<td>019</td>
<td>3025</td>
<td>220200</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.TSEX1P1</td>
<td>TS</td>
<td>020</td>
<td>3025</td>
<td>220200</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.TSEX1P1</td>
<td>TS</td>
<td>021</td>
<td>3025</td>
<td>220200</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.TSEX1P1</td>
<td>TS</td>
<td>022</td>
<td>3025</td>
<td>220200</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.TSEX1P1</td>
<td>TS</td>
<td>023</td>
<td>3025</td>
<td>220200</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.TSEX1P1</td>
<td>TS</td>
<td>024</td>
<td>3025</td>
<td>220200</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.TSEX1P1</td>
<td>IX</td>
<td>001</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>002</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>003</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>004</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>005</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>006</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>007</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>008</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>009</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>010</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>011</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>012</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>013</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>014</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>015</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>016</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>017</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>018</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>019</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>020</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>021</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>022</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>023</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.ICEX1P1</td>
<td>IX</td>
<td>024</td>
<td>6062</td>
<td>698511</td>
<td></td>
</tr>
<tr>
<td>BMC40851I</td>
<td>AFREX01.INEX1P1</td>
<td>IX</td>
<td>000</td>
<td>146498</td>
<td>18028355</td>
<td></td>
</tr>
</tbody>
</table>

BMC40852I TOTAL 364586 40077419

BMC40853I NUMREC EST 364586 AVGRECSZ 109
Example 14: Simulating recovery

This example (Figure 89) illustrates the use of the SIMULATE YES option to invoke recovery simulation. Specifying SIMULATE YES necessitates the use of SIMRCVR TABLESPACE and SIMRBLD INDEX (instead of RECOVER TABLESPACE and REBUILD INDEX).

**NOTE**
Simulation is a feature of the Recovery Management for DB2 solution and requires a valid Recovery Management solution password.

Figure 89  Example 14—JCL using the SIMULATE option

```plaintext
//AFREX18  JOB (PAFR),'EXAMPLE 14',
//     CLASS=Q,NOTIFY=&SYSUID,
//     MSGCLASS=X
/*ROUTE XEQ BMCPLX1
/*JOBPARM SYSAFF=DB2A
/*
//RECOVER EXEC PGM=AFRMAIN,REGION=0M,
//      PARM='DGE,EXAMPLE14,NEW/RESTART,MSGLEVEL(2)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN DD *
//OPTIONS SIMULATE YES
SIMRCVR TABLESPACE BMCDBSMP.BMCTS014
SIMRBLD INDEX (ALL) TABLESPACE BMCDBSMP.BMCTS014
/*
//
```
Example 15: Using MAXKSORT for parallel index rebuilds

This example (Figure 90) illustrates the use of the MAXKSORT option to invoke parallel index sorts and rebuilds to rebuild a partitioned index and multiple nonpartitioned indexes.

Figure 90 Example 15—JCL using the MAXKSORT option for parallel index rebuilds

```plaintext
//AFREX16 JOB (PAFR),"EXAMPLE 15", CLASS=Q,NOTIFY=&SYSUID, MSGCLASS=X,TIME=(0,5) /*ROUTE XEO BMCPLX1 /*JOBPARM SYSAFF=DB2A /*
/*RECOVER EXEC PGM=AFRMAIN,REGION=0M, PARM='DEBA,EXAMPLE15,NEW/RESTART,MSGLEVEL(0)' /*STELIB DD DISP=SHR,DSN=product.libraries /* DD DISP=SHR,DSN=DB2.DSNEXIT /* DD DISP=SHR,DSN=DB2.DSNLOAD /*SYIN DD * OPTIONS MAXKSORT 12 REBUILD INDEX (ALL) TABLESPACE DBRDASTC.TSRDASTC /*
```

Figure 91 through Figure 93 on page 394 show the following output, which is produced by running this JCL:

- AFRSUMRY
- AFRSTMT
- AFRPRINT

See Table 10 on page 277 for a description of each of these files.

Figure 91 Example 15—AFRSUMRY for parallel index rebuilds with MAXKSORT (part 1 of 2)

```
BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40876I MAINT: NO RECOVER PLUS PTFs APPLIED

BMC96010I SOLUTION COMMON CODE V10.1.00
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFs APPLIED

BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276
BMC40876I MAINT: BPJ0279 BPJ0288

BMC40002I UTILITY ID = RDAJBMT.AFR15, DB2 SUBSYSTEM ID = DXW.
BMC40094I ATTEMPTING TO CONNECT TO DB2 SUBSYSTEM DXW, USING PLAN AFRB101T
BMC40342I A GROUP ATTACH WAS PERFORMED WITH DXW
BMC40024I SUCCESSFUL CONNECT TO W4 (RELEASE 910 OF DB2) USING PLAN AFRB101T
BMC40067I DB2 W4 IS MEMBER DXW4 IN DATA SHARING GROUP DSNDXW
BMC40475I INPUT STATEMENTS PRINTED IN AFRSTMT

BMC40475I INPUT STATEMENTS PRINTED IN AFRSTMT

BMC40336I ANALYZE FINISHED.
BMC40287I ELAPSED TIME = 00:00:03, UTILITY ELAPSED TIME = 00:00:03
```
**Example 15: Using MAXKSORT for parallel index rebuilds**

**Figure 91**  
Example 15—AFRSUMRY for parallel index rebuilds with MAXKSORT (part 2 of 2)

<table>
<thead>
<tr>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU TIME</td>
<td>00:00:00</td>
</tr>
<tr>
<td>UTILITY CPU TIME</td>
<td>00:00:00</td>
</tr>
<tr>
<td>PLANNING FINISHED</td>
<td></td>
</tr>
<tr>
<td>ELSAED TIME</td>
<td>00:00:03</td>
</tr>
<tr>
<td>UTILITY ELSAED TIME</td>
<td>00:00:06</td>
</tr>
<tr>
<td>CPU TIME</td>
<td>00:00:00</td>
</tr>
<tr>
<td>UTILITY CPU TIME</td>
<td>00:00:00</td>
</tr>
<tr>
<td>PLAN EXECUTION FINISHED</td>
<td></td>
</tr>
<tr>
<td>ELSAED TIME</td>
<td>00:00:11</td>
</tr>
<tr>
<td>UTILITY ELSAED TIME</td>
<td>00:00:18</td>
</tr>
<tr>
<td>CPU TIME</td>
<td>00:00:00</td>
</tr>
<tr>
<td>UTILITY CPU TIME</td>
<td>00:00:00</td>
</tr>
<tr>
<td>EXECUTION SUMMARY</td>
<td></td>
</tr>
<tr>
<td>INDEXES REBUILT</td>
<td>2</td>
</tr>
</tbody>
</table>

**Figure 92**  
Example 15—AFRSTMT for parallel index rebuilds with MAXKSORT (part 1 of 2)

<table>
<thead>
<tr>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER PLUS FOR DB2 V10.1.00</td>
<td></td>
</tr>
<tr>
<td>COPYRIGHT BMC SOFTWARE INC. 1991-2011</td>
<td></td>
</tr>
<tr>
<td>RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884</td>
<td></td>
</tr>
<tr>
<td>CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO <a href="mailto:SUPPORT@BMC.COM">SUPPORT@BMC.COM</a></td>
<td></td>
</tr>
<tr>
<td><strong>z/OS RELEASE</strong></td>
<td>1.9.0</td>
</tr>
<tr>
<td><strong>PID</strong></td>
<td>HB67740</td>
</tr>
<tr>
<td><strong>SMS RELEASE</strong></td>
<td>1.9.0</td>
</tr>
<tr>
<td>AVAILABLE REGION BELOW 16M:</td>
<td>8836K</td>
</tr>
<tr>
<td>AVAILABLE REGION ABOVE 16M:</td>
<td>1574392K</td>
</tr>
<tr>
<td>NUMBER OF CPUS</td>
<td>3</td>
</tr>
<tr>
<td><strong>RESTART</strong></td>
<td>NEW/RESTART</td>
</tr>
<tr>
<td><strong>MSGLEVEL</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>CHECKPT</strong></td>
<td>PHASE</td>
</tr>
<tr>
<td><strong>CHECKINT</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>RDB2STAT</strong></td>
<td>YES</td>
</tr>
<tr>
<td><strong>AMSCAT</strong></td>
<td>NO</td>
</tr>
<tr>
<td><strong>SMCORE</strong></td>
<td>0K, 0K</td>
</tr>
<tr>
<td><strong>RESINV</strong></td>
<td>OK, OK</td>
</tr>
<tr>
<td><strong>WKUNIT</strong></td>
<td>SYSAUXLDA</td>
</tr>
<tr>
<td><strong>RCLTSK</strong></td>
<td>10</td>
</tr>
<tr>
<td><strong>IXRECP</strong></td>
<td>NO</td>
</tr>
<tr>
<td><strong>MAXDRIVES</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>MAXLOGS</strong></td>
<td>3</td>
</tr>
<tr>
<td><strong>MAXSORT</strong></td>
<td>YES</td>
</tr>
<tr>
<td><strong>KSORTSHARE</strong></td>
<td>YES</td>
</tr>
<tr>
<td><strong>PUBLICPLAN</strong></td>
<td>YES</td>
</tr>
<tr>
<td><strong>BINDQUALIFIER</strong></td>
<td>YES</td>
</tr>
<tr>
<td><strong>BACKOUT</strong></td>
<td>AUTO</td>
</tr>
<tr>
<td><strong>TAPE DEVICE01</strong></td>
<td>3590-1</td>
</tr>
<tr>
<td><strong>TAPE DEVICE02</strong></td>
<td>3490</td>
</tr>
<tr>
<td><strong>TAPE DEVICE03</strong></td>
<td>3480X</td>
</tr>
<tr>
<td><strong>TAPE DEVICE04</strong></td>
<td>ATLVTS</td>
</tr>
<tr>
<td><strong>TAPE DEVICE05</strong></td>
<td>ATL3592</td>
</tr>
</tbody>
</table>

**Figure 91**  
Example 15—AFRSUMRY for parallel index rebuilds with MAXKSORT (part 2 of 2)
Example 15: Using MAXKSORT for parallel index rebuilds

| BMC409671 | TAPE DEVICE06 = CART |
| BMC409671 | TAPE DEVICE07 = CARTVTS |
| BMC409671 | TAPE DEVICE08 = CRT8NS |
| BMC409671 | TAPE DEVICE09 = CRT9NS |
| BMC409671 | TAPE DEVICE10 = DANBARR |
| BMC409671 | TAPE DEVICE11 = EXT3592 |
| BMC409671 | TAPE DEVICE12 = IBMATL |
| BMC409671 | TAPE DEVICE13 = IBMVTS |
| BMC409671 | TAPE DEVICE14 = SA3590 |
| BMC409671 | TAPE DEVICE15 = TAPEABL |
| BMC409671 | TAPE DEVICE16 = 3590B |
| BMC409671 | TAPE DEVICE17 = 3590E |
| BMC409671 | TAPE DEVICE18 = 3400 |
| BMC409671 | TAPE DEVICE19 = 3400-9 |
| BMC409671 | TAPE DEVICE20 = SYS3480R |
| BMC409671 | TAPE DEVICE21 = SYS348XR |

**BMC409291** SORT VALUES: SORTCORE = 0 K (SORT MEMORY VALUE)

**BMC409471** VSCORET = 0 K (TOTAL MEMORY OVERRIDE)

**BMC96111I** DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE

**BMC96113I** COPIES = (LP,LB) (DEFAULT SEQUENCE FOR IMAGE COPIES)

**BMC96114I** ACCUMS = (LP,LB) (DEFAULT SEQUENCE FOR CA FILES)

**BMC96112I** DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE

**BMC96113I** COPIES = (RP,RB) (DEFAULT SEQUENCE FOR IMAGE COPIES)

**BMC96114I** ACCUMS = (RP,RB) (DEFAULT SEQUENCE FOR CA FILES)

**BMC40018I** RECOVER PLUS FOR DB2 V10.1.00

**BMC40876I** MAINT: NO RECOVER PLUS PTFS APPLIED

**BMC96010I** SOLUTION COMMON CODE V10.1.00

**BMC40876I** MAINT: NO SOLUTION COMMON CODE PTFS APPLIED

**BMC40611I** BMCSORT ENGINE V2.3.01

**BMC40876I** MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0276

**BMC40876I** MAINT: BPJ0279 BPJ0288

**BMC40037I** UTILID RDAJBM.AFR16 WAS NOT FOUND IN THE BMCUTIL TABLE - UTILITY WILL RUN AS IF 'NEW' WERE SPECIFIED

**BMC40611I** INPUT STATEMENTS:

**BMC40101I** OPTIONS MAXKSORT 12

**BMC40101I** REBUILD INDEX (ALL) TABLESPACE AFOREX02.TSEX2P32

**BMC40101I**

**BMC409571** MAXKSORT VALUE OVERRIDDEN, NEW VALUE = 12

**BMC40335I** ANALYZE STARTING 01/13/2011 08:41:18

**BMC40335I** CATALOG LOOKUP STARTING 01/13/2011 08:41:18

**BMC40068I** THE KEYS FOR INDEX AFR.INEX2P32 WILL GO DIRECTLY TO THE SORT BECAUSE THERE IS NO SYSUT1 DD STATEMENT IN THE JCL

**BMC40068I** THE KEYS FOR INDEX AFR.ICEX2P32 WILL GO DIRECTLY TO THE SORT BECAUSE THERE IS NO SYSUT1 DD STATEMENT IN THE JCL

**BMC40633I** USING STANDARD DB2 SECURITY

**BMC400705I** VOLUMES IN STOGROUP SYSDFLT WILL BE USED IN THE FOLLOWING ORDER:

**BMC400707I** (<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>)

**BMC40336I** CATALOG LOOKUP FINISHED.

**BMC40287I** ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00

**BMC40288I** CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

**BMC40335I** SYNCHRONIZATION STARTING 01/13/2011 08:41:18

**BMC40336I** SYNCHRONIZATION FINISHED.

**BMC40287I** ELAPSED TIME = 00:00:02, UTILITY ELAPSED TIME = 00:00:03

**BMC40288I** CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

**BMC40335I** PLANNING STARTING 01/13/2011 08:41:21

**BMC40335I** PLAN EXECUTION STARTING 01/13/2011 08:41:24
### Example 15: Using MAXKSORT for parallel index rebuilds

#### Figure 93  Example 15—AFRPRINT for parallel index rebuilds with MAXKSORT (part 1 of 3)

<table>
<thead>
<tr>
<th>BMC40018I</th>
<th>RECOVER PLUS FOR DB2 V10.1.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40473I</td>
<td>COPYRIGHT BMC SOFTWARE INC. 1991-2011</td>
</tr>
<tr>
<td>BMC46173I</td>
<td>RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884</td>
</tr>
<tr>
<td>BMC40474I</td>
<td>CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO <a href="mailto:SUPPORT@BMC.COM">SUPPORT@BMC.COM</a></td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 1</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 2</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 3</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 4</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 5</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 6</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 7</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 8</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 9</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 10</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 11</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 12</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 13</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 14</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 15</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 16</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 17</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 18</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 19</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 20</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 21</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 22</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 23</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 24</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 25</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 26</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 27</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 28</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 29</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 30</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 31</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 32</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 33</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 34</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 35</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 36</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 37</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 38</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 39</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 40</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 41</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 42</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 43</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 44</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 45</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = AFREX02.TSEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 46</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>UNLOAD COMPLETE.</td>
</tr>
</tbody>
</table>
Example 15: Using MAXKSORT for parallel index rebuilds

Figure 93  Example 15—AFRPRINT for parallel index rebuilds with MAXKSORT (part 2 of 3)
Example 15: Using MAXKSORT for parallel index rebuilds

**Figure 93  Example 15—AFRPRINT for parallel index rebuilds with MAXKSORT (part 3 of 3)**

<table>
<thead>
<tr>
<th>BMC40303I</th>
<th>INDEX = AFR.ICEX2P32</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 15</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = AFR.ICEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 20</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = AFR.ICEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 16</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = AFR.ICEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 17</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = AFR.ICEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 18</td>
</tr>
<tr>
<td>BMC40483I</td>
<td>BUILD COMPLETE.</td>
</tr>
<tr>
<td>BMC40303I</td>
<td>INDEX = AFR.ICEX2P32</td>
</tr>
<tr>
<td>BMC40269I</td>
<td>PARTITION = 19</td>
</tr>
</tbody>
</table>
Example 16: Using MAXKSORT and recovering a table space

This example (Figure 94) illustrates the recovery of a table space and the use of the MAXKSORT option to invoke parallel index sorts and rebuilds for the partitioned index and the nonpartitioned indexes.

Figure 94  Example 17—JCL for table space recovery and index recovery using the MAXKSORT option

```
//AFREX17  JOB (PAFR),"EXAMPLE 16",
//         CLASS=Q,NOTIFY=&SYSUID,
//         MSGCLASS=X
/*ROUTE XEQ BMCPLX1
/*JOBPARM SYSAFF=DB2A
/*
//RECOVER EXEC PGM=AFRMAIN,REGION=0M,
//          PARM='DEBA,EXAMPLE16,NEW/RESTART,MSGLEVEL(2)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
//          DD DISP=SHR,DSN=DB2.DSNEXIT
//          DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN     DD *
OPTIONS MAXKSORT 12
RECOVER TABLESPACE DBRDASTC.TSRDASTC
REBUILD INDEX (ALL) TABLESPACE DBRDASTC.TSRDASTC
/*
//```
Example 17: Using timestamp recovery

The inflight resolution technology of the Recovery Management for DB2 solution enables you to perform a consistent recovery to any timestamp or LRSN/RBA. The ability to resolve inflight units of work at any point in time or to any log point completely eliminates the need to perform quiesces to establish consistent recovery points during application execution. The ability to avoid quiesces can dramatically improve the availability of your DB2 data.

The timestamp recovery feature of Recovery Management for DB2 solution uses inflight resolution technology to perform a consistent point-in-time recovery to any user-specified timestamp. The Recovery Management solution translates the timestamp to an RBA or LRSN, recovers the objects, then resolves all inflight units of work for both data sharing and non-data-sharing systems. (The online interface of RECOVERY MANAGER for DB2 supports this feature on data sharing systems.)

NOTE
Timestamp recovery and inflight resolution technology use features from the RECOVERY MANAGER for DB2, RECOVER PLUS for DB2, and Log Master for DB2 components of the Recovery Management for DB2 solution. The Recovery Management solution password is required to use the feature.


Figure 95 is an example of RECOVER PLUS JCL that includes the options for a timestamp recovery.

Figure 95  Example 17—JCL using RECOVERYPOINT for timestamp recovery

```
//AFREX18 JOB (PAPRF), 'EXAMPLE 17',
//         CLASS=Q, NOTIFY=&SYSUID,
//         MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN, REGION=0M,
//             PARM='DGE,TSREC,NEW/RESTART,MSGLEVEL(2)'
//STEPLIB DD DISP=SHR, DSN=product.libraries
// DD DISP=SHR, DSN=DB2.DSNEXIT
// DD DISP=SHR, DSN=DB2.DSNLOAD
//SYSIN DD *
RECOVER TABLESPACE WRPO801.WRPTS01
RECOVER TABLESPACE WRPO801.WRPTS01
RECOVER INDEX(ALL) TABLESPACE WRPO801.WRPTS01
/*
//
```
Example 18: Using the MAXLSORT default value

In this example, RECOVER PLUS does a basic recovery of table space and uses the default value of the MAXLSORT installation option so that RECOVER PLUS uses concurrent log sorts. If you need to override the default value for MAXLSORT, you can specify a MAXLSORT value on the OPTIONS command in the SYSIN.

If indexes were recovered in the job, the default value of the KSORTSHARE installation option (YES) would specify that RECOVER PLUS should spread the key sorts over all tasks. You can also specify KSORTSHARE on the OPTION command.

Figure 96  Example 18—JCL using default values

```
//AFREX18 JOB(PAFR), 'EXAMPLE 18',
//CLASS=Q,NOTIFY=&SYSUID,
//MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN, REGION=0M,
//PARM='&SSID,&UTILID,NEW,MSGLEVEL(2)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD *
RECOVER TABLESPACE G4GLMQDB.V9LMQTS1
/*
```

Figure Figure 97 on page 400 through Figure 102 on page 405 show the following output, which is produced by running this JCL:

- AFRSTMT
- AFRSUMRY
- AFRPRINT
- AFROSUM
- AFRPLN

See Table 10 on page 277 for a description of each of these files.
Example 18: Using the MAXLSORT default value

The AFRSTMT output shown in Figure 97 shows the MAXLSORT value as 50 for this job. The MAXLSORT installation option defaults to 0, which indicates that RECOVER PLUS will determine the value of MAXLSORT for a job. KSORTSHARE is also highlighted in this output.

Figure 97 Example 18—AFRSTMT showing the MAXLSORT value (part 1 of 2)
Chapter 5  Examples of RECOVER PLUS jobs  401

Example 18: Using the MAXLSORT default value

The SYSPICK output in Figure 98 shows that the copies needed from this recovery are available on different volumes. This is important when RECOVER PLUS uses MAXLSORT to allow multiple log sorts. (If all of the copies are on the same volume, there is no advantage to using MAXLSORT.)

Figure 97  Example 18—AFRSTMT showing the MAXLSORT value (part 2 of 2)

| BMC409671 | TAPE DEVICE17 = 3480 |
| BMC409671 | TAPE DEVICE18 = 3400-9 |
| BMC409671 | TAPE DEVICE19 = SYS3480R |
| BMC409671 | TAPE DEVICE20 = SYS348XR |
| BMC40929I | SORT VALUES: SORTCORE = 0K (SORT MEMORY VALUE) |
| BMC40947I | VSCORET = 0K (TOTAL MEMORY OVERRIDE) |
| BMC96111I | DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE |
| BMC96113I | COPIES = (LP,LB) (DEFAULT SEQUENCE FOR IMAGE COPIES) |
| BMC96114I | ACCUMS = (LP,LB) (DEFAULT SEQUENCE FOR CA FILES) |
| BMC96112I | DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE |
| BMC96113I | COPIES = (RP,RB) (DEFAULT SEQUENCE FOR IMAGE COPIES) |
| BMC96114I | ACCUMS = (RP,RB) (DEFAULT SEQUENCE FOR CA FILES) |

Figure 98  Example 18—SYSPICK output

| DATE 01/05/2011 RECOVER PLUS FOR DB2 | PAGE 001 of 001 |
| TIME 13:50:51 | VOLUME PICK LIST |
| - VOLSER | DEV TYPE | DATASET NAME | SEQ |
| 122722 | CART | AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQT51.P01 | 001 |
| 155025 | CART | AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQT51.P02 | 001 |
| 117066 | CART | AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQT51.P02 | 001 |
| 133817 | CART | AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQT51.P03 | 001 |

Figure 99  Example 18—AFRSUMRY (part 1 of 2)

| BMC40018I | RECOVER PLUS FOR DB2 V10.1.00 |
| BMC40473I | COPYRIGHT BMC SOFTWARE INC. 1991-2011 |
| BMC40474I | CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM |

BMC40001I  UTILITY EXECUTION STARTING 01/05/2011 13:50:46

BMC40018I  RECOVER PLUS FOR DB2 V10.1.00
BMC40876I  MAINT: NO RECOVER PLUS PTFS APPLIED
BMC40610I  SOLUTION COMMON CODE V10.1.00
BMC40876I  MAINT: NO SOLUTION COMMON CODE PTFS APPLIED
BMC96010I  BMCSORT ENGINE V2.3.01
BMC40002I  UTILITY ID = V9LMQ. DB2 SUBSYSTEM ID = DEDL.
BMC40002I  UTILITY ID = V9LMQ. DB2 SUBSYSTEM ID = DEDL.
BMC40002I  ATTEMPTING TO CONNECT TO DB2 SUBSYSTEM DEDL. USING PLAN AFRB101T
BMC40002I  SUCCESSFUL CONNECT TO DEDL (RELEASE 910 OF DB2) USING PLAN AFRB101T
BMC40475I  INPUT STATEMENTS PRINTED IN AFRSTMT
BMC40336I  ANALYZE FINISHED.
BMC40287I  ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I  CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40336I  PLANNING FINISHED.
BMC40287I  ELAPSED TIME = 00:00:03, UTILITY ELAPSED TIME = 00:00:04
BMC40288I  CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

Figure 97  Example 18—AFRSTMT showing the MAXLSORT value (part 2 of 2)
Example 18: Using the MAXLSORT default value

The BMC96304I messages highlighted in the AFRPRINT in Figure 100 indicate the use of parallel merges.

**Figure 99  Example 18—AFRSUMRY (part 2 of 2)**

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40336I</td>
<td>PLAN EXECUTION FINISHED.</td>
</tr>
<tr>
<td>BMC40287I</td>
<td>ELAPSED TIME = 00:01:12, UTILITY ELAPSED TIME = 00:01:17</td>
</tr>
<tr>
<td>BMC40288I</td>
<td>CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40761I</td>
<td>EXECUTION SUMMARY</td>
</tr>
<tr>
<td>BMC4077I</td>
<td>TABLE SPACES / PHASES (PARTITIONS) RECOVERED : 1 (3)</td>
</tr>
<tr>
<td>BMC40005I</td>
<td>UTILITY EXECUTION COMPLETE 01/05/2011 13:52:04, RETURN CODE = 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40485I</td>
<td>END OF PHASE SUBTASK MESSAGES</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40018I</td>
<td>RECOVER PLUS FOR DB2 V10.1.00</td>
</tr>
<tr>
<td>BMC4073I</td>
<td>COPYRIGHT BMC SOFTWARE INC. 1991-2011</td>
</tr>
<tr>
<td>BMC40474I</td>
<td>CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO <a href="mailto:SUPPORT@BMC.COM">SUPPORT@BMC.COM</a></td>
</tr>
</tbody>
</table>

**Figure 100  Example 18—AFRPRINT (part 1 of 3)**

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40016I</td>
<td>MERGE PHASE STARTING 01/05/2011 13:50:52</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40484I</td>
<td>MESSAGES FROM PHASE SUBTASK:</td>
</tr>
<tr>
<td>BMC40060I</td>
<td>PENDING FLAG RECP SET FOR</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DEDLCAT.DSNDBC.G4GLMQDB.V9LMQTS1.I0001.A002</td>
</tr>
<tr>
<td>BMC40356I</td>
<td>AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DEDLCAT.DSNDBC.G4GLMQDB.V9LMQTS1.I0001.A002</td>
</tr>
<tr>
<td>BMC40350I</td>
<td>DATA SET DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00033 ELAPSED = 00:00:00</td>
</tr>
<tr>
<td>BMC40485I</td>
<td>END OF PHASE SUBTASK MESSAGES</td>
</tr>
<tr>
<td>BMC40350I</td>
<td>DATA SET AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P02 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00036 ELAPSED = 00:00:01</td>
</tr>
<tr>
<td>BMC4060I</td>
<td>THIS PHASE WILL RUN IN EXECUTION SUBTASK 2</td>
</tr>
<tr>
<td>BMC40593I</td>
<td>START MERGE COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40102I</td>
<td>MERGE TABLESPACE STEP STARTING</td>
</tr>
<tr>
<td>BMC40795I</td>
<td>FULL COPY DATASET IS AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P02</td>
</tr>
<tr>
<td>BMC40799I</td>
<td>MERGE STARTING FOR DATA SET DEDLCAT.DSNDBC.G4GLMQDB.V9LMQTS1.I0001.A002</td>
</tr>
<tr>
<td>BMC40593I</td>
<td>INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40593I</td>
<td>SPIN COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40597I</td>
<td>PROGRESS--PAGE NUMBER 100000, ELAPSED TIME = 00:00:10</td>
</tr>
<tr>
<td>BMC40597I</td>
<td>PROGRESS--PAGE NUMBER 300000, ELAPSED TIME = 00:00:35</td>
</tr>
<tr>
<td>BMC40597I</td>
<td>PROGRESS--PAGE NUMBER 400000, ELAPSED TIME = 00:00:48</td>
</tr>
<tr>
<td>BMC40597I</td>
<td>PROGRESS--PAGE NUMBER 500000, ELAPSED TIME = 00:01:00</td>
</tr>
<tr>
<td>BMC40902I</td>
<td>TOTAL WAITS = 2037, TOTAL WAIT TIME = 00:00:34</td>
</tr>
<tr>
<td>BMC40981I</td>
<td>FULL IC AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P02 READ WAITS = 181, WAIT TIME = 00:00:00</td>
</tr>
<tr>
<td>BMC40905I</td>
<td>SPACE WRITE WAITS = 1856, WAIT TIME = 00:00:33</td>
</tr>
<tr>
<td>BMC40900I</td>
<td>NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 571484, USED = 571484</td>
</tr>
<tr>
<td>BMC40700I</td>
<td>NUMBER OF PAGES READ FROM FULL COPY TOTAL = 571484, USED = 571484</td>
</tr>
<tr>
<td>BMC40798I</td>
<td>NUMBER OF PAGES OUTPUT IS = 571484</td>
</tr>
<tr>
<td>BMC40900I</td>
<td>MERGE TABLESPACE STEP COMPLETE</td>
</tr>
<tr>
<td>BMC40593I</td>
<td>MERGE PROCESS COMPLETE, ELAPSED TIME = 00:01:09</td>
</tr>
<tr>
<td>BMC40352I</td>
<td>DONAME SYS00033 (DSN DEDLCAT.DSNDBC.G4GLMQDB.V9LMQTS1.I0001.A002) WAS SUCCESSFULLY UNALLOCATED</td>
</tr>
<tr>
<td>BMC40352I</td>
<td>DONAME SYS00036 (DSN AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P02) WAS SUCCESSFULLY UNALLOCATED</td>
</tr>
<tr>
<td>BMC40601I</td>
<td>PENDING FLAG RECP RESET FOR</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40091I</td>
<td>01/05/2011 13:52:04 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE</td>
</tr>
</tbody>
</table>
Example 18: Using the MAXLSORT default value

**Figure 100 Example 18—AFRPRINT (part 2 of 3)**

```
BMC40012I MERGE PHASE COMPLETE. ELAPSED TIME = 00:01:11, TIME SINCE UTILITY START = 00:01:17
BMC40868I MERGE PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00

BMC40016I MERGE PHASE STARTING 01/05/2011 13:50:54
BMC40301I TABLESPACE = G4GMQDB.V9LMQTS1
BMC40305I DSNUM = 3

BMC40484I MESSAGES FROM PHASE SUBTASK:

BMC40417I 1 VSAM DATA SETS NOW ALLOCATED
BMC40060I PENDING FLAG RECP SET FOR
BMC40345I SPACE = G4GMQDB.V9LMQTS1
BMC40305I DSNUM = 3
BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET
DEDLCAT.DSNDBC.G4GMQDB.V9LMQTS1.I0001.A003
BMC40356I AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET
DEDLCAT.DSNDBC.G4GMQDB.V9LMQTS1.I0001.A003
BMC40350I DATA SET DEDLCAT.DSNDBC.G4GMQDB.V9LMQTS1.I0001.A003 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00037
ELAPSED = 00:00:01
BMC40417I 3 VSAM DATA SETS NOW ALLOCATED

BMC40486I END OF PHASE SUBTASK MESSAGES

BMC40350I DATA SET AFR.MXLTV9.BMCFCPY2.G4GMQDB.V9LMQTS1.P03 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00038 ELAPSED = 00:00:01
BMC40963I THIS PHASE WILL RUN IN EXECUTION SUBTASK 3

BMC40593I START MERGE COMPLETE, ELAPSED TIME = 00:00:00
BMC40102I MERGE TABLESPACE STEP STARTING
BMC40799I FULL COPY DATASET IS AFR.MXLTV9.BMCFCPY2.G4GMQDB.V9LMQTS1.P03
BMC40799I MERGE STARTING FOR DATA SET DEDLCAT.DSNDBC.G4GMQDB.V9LMQTS1.I0001.A003

BMC40593I INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00
BMC40593I SPIN COMPLETE, ELAPSED TIME = 00:00:00
BMC40597I PROGRESS--PAGE NUMBER 100000, ELAPSED TIME = 00:00:10
BMC40597I PROGRESS--PAGE NUMBER 200000, ELAPSED TIME = 00:00:21
BMC40597I PROGRESS--PAGE NUMBER 300000, ELAPSED TIME = 00:00:35
BMC40597I PROGRESS--PAGE NUMBER 400000, ELAPSED TIME = 00:00:48
BMC40902I TOTAL WAITS = 1550, TOTAL WAIT TIME = 00:00:27
BMC40981I FULL IC AFR.MXLTV9.BMCFCPY2.G4GMQDB.V9LMQTS1.P03 READ WAITS = 164, WAIT TIME = 00:00:00
BMC40905I SPACE WRITE WAITS = 1386, WAIT TIME = 00:00:26
BMC40963I NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 428470, USED = 428470
BMC40798I NUMBER OF PAGES READ FROM FULL COPY TOTAL = 428470, USED = 428470
BMC40798I NUMBER OF PAGES OUTPUT IS = 428470
BMC40798I NUMBER OF PAGES OUTPUT PRINTED = 428470

BMC40593I MESSAGES FROM PHASE SUBTASK:

BMC40963I DDNAME SYS00037 (DSN DEDLCAT.DSNDBC.G4GMQDB.V9LMQTS1.I0001.A003) WAS SUCCESSFULLY UNALLOCATED
BMC40350I DATA SET DEDLCAT.DSNDBC.G4GMQDB.V9LMQTS1.I0001.A003 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00037
ELAPSED = 00:00:00

BMC40060I PENDING FLAG RECP RESET FOR
BMC40345I SPACE = G4GMQDB.V9LMQTS1
BMC40305I DSNUM = 3

BMC40091I 01/05/2011 13:51:48 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE

BMC40016I MERGE PHASE COMPLETE. ELAPSED TIME = 00:00:53, TIME SINCE UTILITY START = 00:01:01
BMC40868I MERGE PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00

BMC40016I MERGE PHASE STARTING 01/05/2011 13:51:48
BMC40301I TABLESPACE = G4GMQDB.V9LMQTS1
BMC40305I DSNUM = 1

BMC40350I DATA SET AFR.MXLTV9.BMCFCPY2.G4GMQDB.V9LMQTS1.P01 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00030 ELAPSED = 00:00:00

BMC40060I PENDING FLAG RECP SET FOR
BMC40345I SPACE = G4GMQDB.V9LMQTS1
BMC40305I DSNUM = 1

BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET
DEDLCAT.DSNDBC.G4GMQDB.V9LMQTS1.I0001.A001
BMC40356I AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET
DEDLCAT.DSNDBC.G4GMQDB.V9LMQTS1.I0001.A001
BMC40350I DATA SET DEDLCAT.DSNDBC.G4GMQDB.V9LMQTS1.I0001.A001 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00035
ELAPSED = 00:00:00

BMC40060I PENDING FLAG RECP SET FOR
```

Chapter 5   Examples of RECOVER PLUS jobs   403
Example 18: Using the MAXLSORT default value

Figure 100  Example 18—AFRPRINT (part 3 of 3)

BMC40931 START MERGE COMPLETE. ELAPSED TIME = 00:00:00
BMC40102 MERGE TABLESPACE STEP STARTING
BMC40795I FULL COPY DATASET IS AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P01
BMC40799I MERGE STARTING FOR DATA SET DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A001
BMC40593I INITIALIZATION COMPLETE. ELAPSED TIME = 00:00:00
BMC40593I SPIN COMPLETE, ELAPSED TIME = 00:00:18
BMC40597I PROGRESS--PAGE NUMBER 100000, ELAPSED TIME = 00:00:09
BMC40597I PROGRESS--PAGE NUMBER 200000, ELAPSED TIME = 00:00:45
BMC40597I PROGRESS--PAGE NUMBER 300000, ELAPSED TIME = 00:00:32
BMC40597I PROGRESS--PAGE NUMBER 400000, ELAPSED TIME = 00:00:45
BMC40902I TOTAL WAITS = 1621, TOTAL WAIT TIME = 00:00:24
BMC40981I FULL IC AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P01 READ WAITS = 215, WAIT TIME = 00:00:00
BMC40905I SPACE WRITE WAITS = 1406, WAIT TIME = 00:00:23
BMC40985I NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 428613, USED = 428613
BMC40790I NUMBER OF PAGES READ FROM FULL COPY TOTAL = 428613, USED = 428613
BMC40798I NUMBER OF PAGES OUTPUT IS = 428613
BMC40900I MERGE TABLESPACE STEP COMPLETE
BMC40593I MERGE PROCESS COMPLETE. ELAPSED TIME = 00:00:48
BMC40352I DDNAME SYS00039 (DSN DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A001) WAS SUCCESSFULLY UNALLOCATED
BMC40352I DDNAME SYS00030 (DSN AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P01) WAS SUCCESSFULLY UNALLOCATED
BMC40060I PENDING FLAG RECP RESET FOR
BMC40345I    SPACE = G4GLMQDB.V9LMQTS1
BMC40305I       DSNUM = 1
BMC40091I 01/05/2011 13:51:42 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE
BMC40012I MERGE PHASE COMPLETE. ELAPSED TIME = 00:00:50, TIME SINCE UTILITY START = 00:00:55
BMC40868I MERGE PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00

Figure 101  Example 18—AFROSUM (part 1 of 2)

BMC40018I RECOVER PLUS FOR DB2 V10.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011
BMC36173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40709I OBJECT SUMMARY:
BMC40301I    TABLESPACE = G4GLMQDB.V9LMQTS1
BMC40713I         PARTITION = 1 OF 3
BMC40711I       DBID = X'02FC'  PSID = X'0002'
BMC40712I       ORIGINAL STATUS = RW
BMC40716I          STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000
BMC40723I          ENCODING SCHEME = EBCDIC

BMC40820I TABLESPACE RECOVERY
BMC40827I LOG RESOURCE RANGE:
BMC40586I FROM LOGPOINT = X'03B2DA266BB2' TO LOGPOINT = X'03B2DA27BED7'

BMC40840I INPUT COPY / SYSCOPY DATA:
BMC40301I 2011-01-06 22.12.26.1069 RBA/LSRN X'03B2DA266BB2' FU
BMC40302I DSN = AFR.MXLTSRT.BMCFCPY2.G4GLMQDB.V9LMQTS1.P01
BMC40641I SHRLEVEL = REFERENCE SITESTYPE = LP DSNUM = 1
BMC40640I 2011-01-06 22.12.04.4922 RBA/LSRN X'03B2DA266BB2' QU

BMC40709I OBJECT SUMMARY:
BMC40301I    TABLESPACE = G4GLMQDB.V9LMQTS1
BMC40713I         PARTITION = 2 OF 3
BMC40711I       DBID = X'02FC'  PSID = X'0002'
BMC40712I       ORIGINAL STATUS = RW
BMC40716I          STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000
BMC40723I          ENCODING SCHEME = EBCDIC

BMC40820I TABLESPACE RECOVERY
**Example 18: Using the MAXLSORT default value**

**Figure 101 Example 18 —AFROSUM (part 2 of 2)**

<table>
<thead>
<tr>
<th>BMC40827I</th>
<th>LOG RESOURCE RANGE:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'03B2DA266B82' TO LOGPOINT = X'03B2DA27BE07'</td>
</tr>
</tbody>
</table>

**Figure 102 Example 18 —AFROSUM (part 1 of 2)**

<table>
<thead>
<tr>
<th>BMC40018I</th>
<th>RECOVER PLUS FOR DB2 V10.1.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40473I</td>
<td>COPYRIGHT BMC SOFTWARE INC. 1991-2011</td>
</tr>
<tr>
<td>BMC40474I</td>
<td>CONTACT BMC SUPPORT AT 1-800-537-1815 OR EMAIL TO <a href="mailto:SUPPORT@BMC.COM">SUPPORT@BMC.COM</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40409I</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 1 OF 3</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = 'X'02FC' PSID = 'X'0002'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40820I</th>
<th>TABLESPACE RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'03B2DA266B82' TO LOGPOINT = X'03B2DA27BE07'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40409I</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 2 OF 3</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = 'X'0278' PSID = 'X'0002'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40820I</th>
<th>TABLESPACE RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'080C1182A555' TO LOGPOINT = X'080C12B6B9F9'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40409I</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 2 OF 3</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = 'X'0278' PSID = 'X'0002'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40820I</th>
<th>TABLESPACE RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'080C1182A555' TO LOGPOINT = X'080C12B6B9F9'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40409I</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 2 OF 3</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = 'X'0278' PSID = 'X'0002'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40820I</th>
<th>TABLESPACE RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'080C1182A555' TO LOGPOINT = X'080C12B6B9F9'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40409I</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 2 OF 3</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = 'X'0278' PSID = 'X'0002'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40820I</th>
<th>TABLESPACE RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'080C1182A555' TO LOGPOINT = X'080C12B6B9F9'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40409I</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 2 OF 3</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = 'X'0278' PSID = 'X'0002'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40820I</th>
<th>TABLESPACE RECOVERY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40827I</td>
<td>LOG RESOURCE RANGE:</td>
</tr>
<tr>
<td>BMC40585I</td>
<td>FROM LOGPOINT = X'080C1182A555' TO LOGPOINT = X'080C12B6B9F9'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMC40409I</th>
<th>OBJECT SUMMARY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40713I</td>
<td>PARTITION = 2 OF 3</td>
</tr>
<tr>
<td>BMC40711I</td>
<td>DBID = 'X'0278' PSID = 'X'0002'</td>
</tr>
<tr>
<td>BMC40712I</td>
<td>ORIGINAL STATUS = RW</td>
</tr>
<tr>
<td>BMC40716I</td>
<td>STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000</td>
</tr>
<tr>
<td>BMC40723I</td>
<td>ENCODING SCHEME = EBCDIC</td>
</tr>
</tbody>
</table>
Example 18: Using the MAXLSORT default value

**Figure 102  Example 18—AFROSUM (part 2 of 2)**

<table>
<thead>
<tr>
<th>BMC40640I</th>
<th>2011-01-05 13.48.29.8534 RBA/LRSN X'080C1182A555' QUIESCE UTILITY POINT - WRITE(YES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40709I OBJECT SUMMARY:</td>
<td></td>
</tr>
<tr>
<td>BMC40301I TABLESPACE = G4GLMQDB.V9LMQTS1</td>
<td></td>
</tr>
<tr>
<td>BMC40713I PARTITION = 3 OF 3</td>
<td></td>
</tr>
<tr>
<td>BMC40711I DBID = X'0028'  PSID = X'0002'</td>
<td></td>
</tr>
<tr>
<td>BMC40712I ORIGINAL STATUS = RW</td>
<td></td>
</tr>
<tr>
<td>BMC40716I STOGROUP = G4GLMOSG PRIQTY = 25000 SECQTY = 50000</td>
<td></td>
</tr>
<tr>
<td>BMC40723I ENCODING SCHEME = EBCDIC</td>
<td></td>
</tr>
<tr>
<td>BMC40820I TABLESPACE RECOVERY</td>
<td></td>
</tr>
<tr>
<td>BMC40827I LOG RESOURCE RANGE:</td>
<td></td>
</tr>
<tr>
<td>BMC40585I FROM LOGPOINT = X'080C1182A555'  TO LOGPOINT = X'080C12B6B9F9'</td>
<td></td>
</tr>
<tr>
<td>BMC40840I INPUT COPY / SYSCOPY DATA:</td>
<td></td>
</tr>
<tr>
<td>BMC40640I 2011-01-05 13.50.40.3781 RBA/LRSN X'080C1182A555' FULL IMAGE COPY</td>
<td></td>
</tr>
<tr>
<td>BMC40302I DSN = AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P03</td>
<td></td>
</tr>
<tr>
<td>BMC40641I SHRLEVEL = REFERENCE SITETYPE = LP DSNUM = 3</td>
<td></td>
</tr>
<tr>
<td>BMC40640I 2011-01-05 13.48.29.8534 RBA/LRSN X'080C1182A555' QUIESCE UTILITY POINT - WRITE(YES)</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 103  Example 18—AFRPLAN (part 1 of 3)**

| BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT 7,133,884 |
| BMC40018I RECOVER PLUS FOR DB2 V10.1.00 |
| BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2011 |
| BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM |
| BMC40700I EXECUTION PLAN SUMMARY: |
| BMC40701I PHASE: MERGE |
| BMC40721I PHASE INITIALIZATION: |
| BMC40301I TABLESPACE = G4GLMQDB.V9LMQTS1 |
| BMC40305I DSNUM = 1 |
| BMC40738I ALLOCATE INPUT IMAGE COPY FILE |
| BMC40302I DSN = AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P01 |
| BMC40738I FILE OPTIMIZATION |
| BMC40738I ALLOCATE 2 INPUT FILE BUFFER(S) |
| BMC40738I OPEN IMAGE COPY FILE |
| BMC40302I DSN = AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P01 |
| BMC40738I ALLOCATE PAGE CHECK BUFFER (4096 BYTES) |
| BMC40738I ALLOCATE SPACEMAP BUFFER (4096 BYTES) |
| BMC40738I ALLOCATE PASTE BUFFER (4096 BYTES) |
| BMC40738I SET RECOVERY PENDING (RECP) FLAG |
| BMC40738I CREATE SPACE |
| BMC40345I SPACE = G4GLMQDB.V9LMQTS1 |
| BMC40305I DSNUM = 1 |
| BMC40738I ALLOCATE SPACE (READ/WRITE) |
| BMC40345I SPACE = G4GLMQDB.V9LMQTS1 |
| BMC40305I DSNUM = 1 |
| BMC40738I CONNECT TO SPACE |
| BMC40738I ALLOCATE PAGE CHECK BUFFER (4096 BYTES) |
| BMC40738I FILE OPTIMIZATION |
| BMC40738I ALLOCATE 2 SPACE OUTPUT BUFFER(S) |
| BMC40738I ALLOCATE SPACEMAP BUFFER (4096 BYTES) |
| BMC40738I ALLOCATE PASTE BUFFER (4096 BYTES) |
| BMC40702I STEP: MERGE SPACE |
| BMC40738I MERGE SPACE |
| BMC40301I TABLESPACE = G4GLMQDB.V9LMQTS1 |
| BMC40305I DSNUM = 1 |
| BMC40735I PHASE TERMINATION: |
| BMC40738I DISCONNECT FROM SPACE |
| BMC40738I DEALLOCATE SPACE |
### Example 18—AFRPLAN (part 2 of 3)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40345I</td>
<td>SPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 1</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CLOSE IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P01</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 INPUT FILE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE INPUT IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P02</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
<tr>
<td>BMC40701I</td>
<td>PHASE: MERGE</td>
</tr>
<tr>
<td>BMC40721I</td>
<td>PHASE INITIALIZATION:</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE INPUT IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P02</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>ALLOCATE 2 INPUT FILE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>OPEN IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P02</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CREATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>CONNECT TO SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>ALLOCATE SPACE (READ/WRITE)</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>SPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>ALLOCATE PAGE CHECK BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>FILE OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>ALLOCATE 2 SPACE OUTPUT BUFFER(S)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>ALLOCATE SPACEMAP BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>ALLOCATE PASTE BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40702I</td>
<td>STEP: MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>Merging Space</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>PHASE TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DISCONNECT FROM SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE SPACE</td>
</tr>
<tr>
<td>BMC40345I</td>
<td>SPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 2</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE 2 OUTPUT SPACE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PAGE CHECK BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE PASTE BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE SPACEMAP BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>CLOSE IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P02</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>FREE 2 INPUT FILE BUFFER(S)</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DEALLOCATE INPUT IMAGE COPY FILE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>SET RECOVERY PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD RESTART INFORMATION</td>
</tr>
</tbody>
</table>
**Example 18: Using the MAXLSORT default value**

<table>
<thead>
<tr>
<th>BMC40701I</th>
<th>PHASE: MERGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40721I</td>
<td>PHASE_INITIALIZATION:</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE_INPUT_IMAGE_COPY_FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P03</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FILE_OPTIMIZATION</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE_2_INPUT_FILE_BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>OPEN_IMAGE_COPY_FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P03</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE PAGE_CHECK_BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE_SPACEMAP_BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>ALLOCATE_PASTE_BUFFER (4096 BYTES)</td>
</tr>
<tr>
<td>BMC40702I</td>
<td>STEP: MERGE SPACE</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>MERGE SPACE</td>
</tr>
<tr>
<td>BMC40301I</td>
<td>TABLESPACE = G4GLMQDB.V9LMQTS1</td>
</tr>
<tr>
<td>BMC40305I</td>
<td>DSNUM = 3</td>
</tr>
<tr>
<td>BMC40735I</td>
<td>PHASE_TERMINATION:</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE_PAGE_CHECK_BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE_SPACEMAP_BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE_PASTE_BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE_OUTPUT_SPACE_BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE_PAGE_CHECK_BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE_PASTE_BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE_SPACEMAP_BUFFER</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>FREE_INPUT_FILE_BUFFER(S)</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>DEALLOCATE_INPUT_IMAGE_COPY_FILE</td>
</tr>
<tr>
<td>BMC40302I</td>
<td>DSN = AFR.MXLTV9.BMCFCPY2.G4GLMQDB.V9LMQTS1.P03</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RESET_RECOVERY_PENDING (RECP) FLAG</td>
</tr>
<tr>
<td>BMC40738I</td>
<td>RECORD_RESTART_INFORMATION</td>
</tr>
</tbody>
</table>

---

**Figure 103 Example 18—AFRPLAN (part 3 of 3)**
This chapter describes advanced RECOVER PLUS topics.

Overview .................................................. 410
Recovery of partitioned table spaces and indexes ........................................ 410
  Recovering multiple table space partitions in separate jobs ..................... 411
  Recovering table space partitions and rebuilding a partitioned index in multiple jobs .................................................. 413
Rebuilding nonpartitioned indexes with single and multiple jobs ............... 414
Recovering a partitioned table space and rebuilding a nonpartitioned index .................. 416
Recovering a table space and rebuilding a partitioned index and a nonpartitioned index .................................................. 417
Understanding the LOGSORT strategy .................................................. 418
  Why is it faster to sort the log? .................................................. 418
  What are the restrictions when using LOGSORT? ................................ 419
  Does LOGSORT have other advantages or disadvantages? ...................... 419
Using LOGONLY or LOGAPPLY ONLY with point-in-time recoveries ........... 421
Understanding the NOWORKDDN strategy .......................................... 422
Understanding the UNLOADKEYS and BUILDINDEX strategy .................. 423
Using SHRLEVEL CHANGE full copies for recovery ............................. 424
  Attempting to recover to a SHRLEVEL CHANGE copy ......................... 424
Fallback recovery with RECOVER PLUS ........................................... 424
Strategies for point-in-time recovery .................................................. 426
  Using BACKOUT .................................................. 426
  Using Instant Snapshots .................................................. 429
Working with ASCII and Unicode data .............................................. 433
Managing sort performance .................................................. 433
  Recommendations for a large number of concurrent sorts ...................... 434
  Sort file size estimation .................................................. 435
  RECOVER PLUS sort parameters .................................................. 437
  Allocation of sort work space .................................................. 443
Strategies for using copies .................................................. 444
  Image copy data set contents .................................................. 444
  Cabinet copy data set contents .................................................. 444
  Using copies containing multiple data sets ........................................ 444
  Using data sets stacked on tape .................................................. 445
  Using image copies and log records for indexes .................................. 446
Overview

This chapter describes advanced topics including the optimization process that is used by RECOVER PLUS to determine the best approach for recovery and how you can affect performance by specifying particular options or by changing data set allocations.

Recovery of partitioned table spaces and indexes

You can use RECOVER PLUS to recover table space partitions, partitioned indexes, and nonpartitioned indexes in a variety of ways. This section illustrates the following strategies:

- recovering table space partitions in separate jobs with all jobs running concurrently
• recovering table space partitions and recovering or rebuilding the partitioned indexes on the same partitions with all jobs running concurrently

• recovering or rebuilding a nonpartitioned index on multiple partitions in a single job and in multiple jobs

• rebuilding a nonpartitioned index on multiple partitions in separate jobs with all unload keys jobs running concurrently

• recovering table space partitions and recovering or rebuilding the related partitioned and nonpartitioned indexes with multiple RECOVER and REBUILD command statements

Recovering multiple table space partitions in separate jobs

RECOVER PLUS provides the maximum possible concurrency for recovering multiple partitions of a partitioned table space in separate jobs. Figure 104 on page 412 shows multiple jobs concurrently recovering the multiple partitions of a partitioned table space. You can submit as many jobs as required for the number of partitions that you have created. However, if archive log tapes are required for the table space recovery, only two jobs may use a particular log range concurrently because there are at most two copies of the archive log. If your archive logs are stored on DASD, this constraint does not apply. Jobs that encounter other users on both tape copies of an archive log must wait for a copy to become available.
Recovering multiple table space partitions in separate jobs

Figure 104  Multiple jobs recovering partitions of a table space
Recovering table space partitions and rebuilding a partitioned index in multiple jobs

You can rebuild a partitioned index for a partitioned table space in separate jobs. Figure 105 shows multiple jobs recovering the table space partitions and rebuilding the partitions of a partitioned index for a partitioned table. Using separate jobs for the partitions is ideal when there is little or no log to process and the image copies that you have made are on separate data sets.

With sufficient system resources, these jobs may run in less elapsed time than one job takes to recover all of the partitions and rebuild the partitioned index.

Figure 105  Recovering a table space and rebuilding a partitioned index
Rebuilding nonpartitioned indexes with single and multiple jobs

If nonpartitioned indexes are defined on a partitioned table space, you must extract the keys from all partitions of the table space to rebuild the index. You can do this with a single command statement and job as shown in Figure 106.

Figure 106  Rebuilding a nonpartitioned index with a single job
You can also use RECOVER PLUS to extract the keys in separate jobs. The index can then be re-created from the key data sets. Figure 107 shows keys being extracted from a three-partition table space in three jobs followed by a job to build the index. The sorting is in the UNLOADKEYS jobs.

Figure 107 Rebuilding a nonpartitioned index with multiple jobs
Recovering a partitioned table space and rebuilding a nonpartitioned index

If you need to recover a partitioned table space and a nonpartitioned index, you can combine RECOVER TABLESPACE and RECOVER UNLOADKEYS command statements. As shown in Figure 108, you can set up multiple jobs to do the processing on the individual partitions followed by a RECOVER BUILDINDEX job to build the nonpartitioned indexes. If your system has enough resources, these jobs can extract the keys from several partitions concurrently, and may run in less time than running one job to read all of the partitions. Contention for archive log tapes may cause jobs to wait.

Figure 108  Recovering a partitioned table space and a rebuilding nonpartitioned index
When you run a RECOVER BUILDINDEX job, the JCL must include all of the data sets containing the extracted keys, and use a separate SKEYnn ddname for the file from each UNLOADKEYS job step.

### NOTE

When you run a RECOVER BUILDINDEX job, ensure that no updates or other changes have been made to the table space since the keys were extracted. Otherwise, the integrity of the data cannot be guaranteed. For a discussion, see “Concurrency with RECOVER UNLOADKEYS and RECOVER BUILDINDEX” on page 89.

---

### Recovering a table space and rebuilding a partitioned index and a nonpartitioned index

The LOGSORT/MERGE strategy for recovering a table space allows you to concurrently extract the keys for nonpartitioned indexes and rebuild one or more partitioned indexes. After running these concurrent jobs, you can build the nonpartitioned index. The following combination of RECOVER TABLESPACE, REBUILD INDEX, and RECOVER UNLOADKEYS command statements illustrates this method:

1. Run jobs 1 and 2 concurrently.

   **JOB 1:**

   ```
   RECOVER TABLESPACE X.Y DSNUM 1
   REBUILD INDEX (X.YCLUS) PART 1
   RECOVER UNLOADKEYS (X.YNCLUS) PART 1
   ```

   **JOB 2:**

   ```
   RECOVER TABLESPACE X.Y DSNUM 2
   REBUILD INDEX (X.YCLUS) PART 2
   RECOVER UNLOADKEYS (X.YNCLUS) PART 2
   ```

2. When jobs 1 and 2 are complete, run job 3.

   **JOB 3:**

   ```
   RECOVER BUILDINDEX (X.YNCLUS)
   ```
Because RECOVER PLUS reads all of the contents of SYSIN before deciding on a recovery plan, the utility can process the RECOVER TABLESPACE, REBUILD INDEX, and RECOVER UNLOADKEYS command statements for each run as a single unit.

**NOTE**
When you run a RECOVER BUILDINDEX job, ensure that no updates or other changes have been made to the table space since the keys were extracted. Otherwise, the integrity of the data cannot be guaranteed. For a discussion, see “Concurrency with RECOVER UNLOADKEYS and RECOVER BUILDINDEX” on page 89.

For more information, see “Understanding the LOGSORT strategy” and “Understanding the UNLOADKEYS and BUILDINDEX strategy” on page 423.

---

**Understanding the LOGSORT strategy**

Sorting the log allows RECOVER PLUS to use a LOGSORT/MERGE strategy.

RECOVER PLUS examines the pertinent log ranges and extracts the appropriate log records from the DB2 active and archive logs. RECOVER PLUS then passes these log records to BMCSORT, where they are sorted by object, page number, and log point.

Sorting the log allows the objects to be processed sequentially, which is more efficient than accessing them randomly as in a conventional log apply process. Using the LOGSORT/MERGE strategy can reduce the I/O required by performing the following tasks:

- combining the MERGE and LOG APPLY phases
- ensuring that each page is accessed only once

Sorting the log enables you to read multiple log files in parallel, which can reduce the elapsed time necessary to read all of the log records for the run.

**Why is it faster to sort the log?**

Sorting the log records in page number order allows RECOVER PLUS to invoke a merge process that takes a full copy, any number of incremental copies (limited by MAXDRIVES, see page 120), an input change accumulation file, and the sorted log as input and merges them into a recovered space image. In data processing terms, this process is analogous to a classic master file update. The full copy is similar to the
master file because it was the exact image of the table space at some point in time (this is not strictly true for SHRLEVEL CHANGE copies, which are discussed later). The incremental copies, change accumulation files, and log records are like update files containing transactions to be applied to the master file.

The merge process simply applies the changes to the full copy to produce an updated image. This strategy is much faster than the conventional strategy, which first writes a space to the merged image of the full copy and incremental copies, and applies the unsorted log record updates to the space (a random read-write operation). Many conventional recoveries have long log apply phases.

Because the inputs are in page order for the LOGSORT/MERGE strategy, RECOVER PLUS can create the final version of the page in memory and write the table space once as a sequential write operation. Sequential write processes, when appropriately blocked, buffered, and scheduled, are much faster than random write processes.

Because the inputs are in page order for the LOGSORT/MERGE strategy, RECOVER PLUS can create the final version of the page in memory and write the table space once as a sequential write operation. Sequential write processes, when appropriately blocked, buffered, and scheduled, are much faster than random write processes.

**NOTE**

If your full copies are made inline with the IBM DB2 utilities, RECOVER PLUS must first restore the copy to the space before merging other resources. This action may cause more lengthy recoveries.

When indexes are recovered by using copies and log records, they are handled in the same way as table spaces and can benefit from the LOGSORT/MERGE strategy.

### What are the restrictions when using LOGSORT?

Merges that fail cause LOG INPUT to be repeated on restart for log records of the failed merges.

Failure of an image copy that is detected during merge may cause a merge to require additional log records. When additional log records are required, the job must be restarted to complete failed merges.

### Does LOGSORT have other advantages or disadvantages?

The LOGSORT strategy can provide other advantages, as follows:

- If keys are being extracted in a table space recovery because of either a REBUILD INDEX or RECOVER UNLOADKEYS request for an index on a table in the table space, RECOVER PLUS can extract these keys during the merge process of a LOGSORT strategy. In some cases, the keys are sent directly to a sort at this point by using the NOWORKDDN option (see page 422), or the keys are written to the temporary data set specified by the WORKDDN parameter, usually SYSUT1 for
Does LOGSORT have other advantages or disadvantages?

REBUILD INDEX, or the SKEYDDN parameter for RECOVER UNLOADKEYS, depending on how you have set your index sort option. By extracting the keys during the merge process, RECOVER PLUS avoids rereading the table space to rebuild the index, as is the case in the conventional strategy. This strategy can save elapsed time.

- Because RECOVER PLUS sees each page in its final form and writes the object sequentially, very little overhead is incurred by simultaneously making image copies. Therefore, with the LOGSORT strategy, RECOVER PLUS can optionally make and register (in SYSIBM.SYSCOPY) up to four copies of a recovered table space.

- The LOGSORT strategy allows the use of input change accumulation files and the simultaneous processing of a RECOVER command statement and an ACCUM command statement for the same object. (For more information about ACCUM, see the R+/CHANGE ACCUM for DB2 User Guide.)

- The LOGSORT strategy allows the use of BACKOUT for point-in-time recovery. This strategy can be considerably faster than when copies are used. (For more information, see “Strategies for point-in-time recovery” on page 426.)

- The LOGSORT strategy allows RECOVER PLUS to read multiple log files concurrently, which may significantly improve elapsed time. (For more information, see “Reading multiple log files concurrently” on page 451.)

The LOGSORT strategy can provide disadvantages, as follows:

- The potential disadvantage of using the LOGSORT strategy is that it may require additional resources if there are a significant number of log records to sort. This situation can be alleviated by the periodic running of jobs to build change accumulation files by using R+/CHANGE ACCUM. For more information about change accumulation files, see the R+/CHANGE ACCUM for DB2 User Guide.

- RECOVER PLUS invokes BMCSORT. The amount and type of resources required for the sort depend on the number of log records to be sorted.

If you are reading many log files concurrently, the memory usage of the run will increase. Approximately two megabytes of memory (in the extended, private area above the 16-MB line) will be used for each log file that is read concurrently.

You can control the number of log files read concurrently with the MAXLOGS option. For more information about the additional resources that may be required for the LOGSORT strategy, see “Log sort space requirements” on page 290, “Managing sort performance” on page 433, and “Setting the SMCORE option” on page 437.
Using LOGONLY or LOGAPPLY ONLY with point-in-time recoveries

When a point-in-time recovery is performed, two log points are recorded in SYSIBM.SYSCOPY or BMCXCOPY:

- PIT_RBA: the point to which the table space is recovered
- START_RBA: the point at which the point-in-time recovery was performed

**NOTE**

If you are operating in a data sharing environment, log record sequence number (LRSN) values are stored in SYSIBM.SYSCOPY. LRSNs for PIT_RBA indicate the ending point to which the table space is recovered. LRSNs for START_RBA indicate the point at which the point-in-time recovery was performed.

If the log range used for a LOGONLY or LOGAPPLY ONLY recovery includes both the PIT_RBA and the START_RBA from a point-in-time recovery, the log range between those log points is ignored (see Figure 109).

**Figure 109** How PIT_RBAs and registered log points are used during LOGONLY or LOGAPPLY ONLY recoveries

The FROMLOGPOINT of a LOGONLY recovery is obtained from the log point that is stored in the header page, while LOGAPPLY ONLY obtains it from the syntax.

When the FROMLOGPOINT of a LOGONLY or LOGAPPLY ONLY recovery is greater than the PIT_RBA of a point-in-time recovery and the TORBA of the LOGONLY or LOGAPPLY ONLY recovery is greater than or equal to the START_RBA of the point-in-time recovery (see Figure 110 on page 422), RECOVER PLUS does not allow a recovery to be performed. If RECOVER PLUS attempted this recovery, the restored copy may include some, but not all, updates that are located...
between the PIT_RBA and the START_RBA. Recovery of the log records found after the START_RBA depends on none of those updates being present. If you must perform such a recovery and you are using a non-DB2 backup, you must select a backup before the PIT_RBA.

**Figure 110** When a recovery is not allowed by RECOVER PLUS

<table>
<thead>
<tr>
<th>PIT_RBA</th>
<th>FROMLOGPOINT</th>
<th>START_RBA</th>
<th>TORBA</th>
</tr>
</thead>
<tbody>
<tr>
<td>log point X'12FF000'</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>log point X'1312000'</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>log point X'14CB000'</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>log point X'15FF000'</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

JO 1:

```
RECOVER TABLESPACE X,Y INDEP OUTSPACE
   MODEL &VCAT.DSNDBC.&DB.&SP.I001.&DSNUM
   TOCOPY LASTCOPY
```

JO 2:

```
RECOVER TABLESPACE X,Y FROMLOGPOINT LASTCOPY
   LOGAPPLY ONLY
```

**Understanding the NOWORKDDN strategy**

In a conventional REBUILD INDEX strategy that uses WORKDDN, the keys to build the index are written to a work data set (typically SYSUT1) while they are being extracted from the table space data. When the extraction is complete, the keys are read back in and sorted, and then the index is built from the sorted key/RID pairs.

However, using the NOWORKDDN option allows the extracted keys to go directly into a sort process. This strategy avoids writing and rereading the key work data set, and no work data set is required to hold the extracted keys. While restartability may be affected (see Table 12 on page 292), the I/O reduction significantly reduces the time that is required to rebuild the index.
Understanding the UNLOADKEYS and BUILDINDEX strategy

The rebuilding of large nonpartitioned indexes on partitioned table spaces is a major recovery issue. To recover these objects with a conventional strategy requires a sequential read pass of all partitions of the table space. In some cases, the amount of time that is required to read the partitioned space and rebuild the index with a conventional strategy is prohibitive. To facilitate the recovery of these spaces, RECOVER PLUS provides concurrency in the extraction of keys from multiple partitions.

For example, if you have a large 16-partition table space with a nonpartitioned index that must be recovered (perhaps because of media failure), RECOVER PLUS provides an alternative strategy that uses the RECOVER UNLOADKEYS and RECOVER BUILDINDEX command statements. You can submit up to 16 jobs to extract keys from multiple partitions simultaneously. Sixteen jobs are not required. You can submit up to one job per partition. If you have only eight available initiators, you can submit eight jobs, with each job reading two partitions sequentially. Any combination (10-2-4, 4-4-4-4, and so on) of partitions per job is acceptable.

Specify the RECOVER UNLOADKEYS command statement for the nonpartitioned index and specify the PART option to indicate which table space partition is to be used for this key extraction. Include a DD statement in the JCL to specify the work data set to hold the keys. The extracted keys files are sorted and are then input to the RECOVER BUILDINDEX job where the keys are merged to build the index.

The improved concurrency can drastically reduce the elapsed time that is required to rebuild a nonpartitioned index. Because RECOVER UNLOADKEYS performs the sorting, particularly for a large, partitioned space, you may see performance improvements because a number of smaller sorts are done.

You can extract the keys for RECOVER UNLOADKEYS indexes during the table space recovery process, further reducing total elapsed time for combined table space and index recovery. For an example of a combined LOGSORT/MERGE strategy and UNLOADKEYS/BUILDINDEX strategy, see “Example 6: Extracting nonpartitioned index keys” on page 365.
Using SHRLEVEL CHANGE full copies for recovery

You can make a SHRLEVEL CHANGE image copy (“fuzzy” copy) of a table space while updates to the table space are in progress. Using this type of image copy is not a problem during recovery because DB2 RECOVER and RECOVER PLUS use the log point recorded for the SHRLEVEL CHANGE copy, which is the log point before the start of the copy. This log point is the point at which the recover utilities begin examining log records. All log records that might need to be applied to the table space are examined. Because each DB2 page contains the log point of the last update to the page, both utilities can determine whether a log update really needs to be applied to the page or if that update is already reflected in the current page image.

Attempting to recover to a SHRLEVEL CHANGE copy

With RECOVER PLUS, you can specify RECOVER TABLESPACE TOCOPY that uses a SHRLEVEL CHANGE copy. However, the data on such a copy may not be consistent. This procedure is not recommended and RECOVER PLUS issues a warning to this effect. For more information, see “TOCOPY specification” on page 185.

Fallback recovery with RECOVER PLUS

Fallback in RECOVER PLUS describes the use of an equivalent or prior image copy when the image copy originally chosen for recovery is not available or is invalid. In most cases, RECOVER PLUS can fall back to another copy automatically. However, in a few cases where automatic fallback does not occur, it is necessary to restart a failed job.

**NOTE**

RECOVER PLUS supports automatic fallback for the following copies:

- standard image copies and Instant Snapshot copies
- full and incremental copies
- copies of table spaces or index spaces
Automatic fallback processing includes the following behavior for RECOVER PLUS:

- During the ANALYZE phase, if RECOVER PLUS detects that the specified image copy is not available, RECOVER PLUS falls back to an equivalent image copy if one is available. If such an image copy is not available, RECOVER PLUS falls back to an earlier image copy. If no prior image copy is available, RECOVER PLUS tries to recover from the log. If the required log is no longer available, fallback fails and the space is unrecoverable.

- If the specified image copy is unavailable when RECOVER PLUS attempts to allocate the copy, the recovery falls back to an equivalent image copy. With automatic fallback, RECOVER PLUS uses an earlier image copy or LOAD REPLACE LOG YES or REORG LOG YES log point to recover the space.

- If the selected image copy is available but is corrupted, RECOVER PLUS does not detect this until after image copy allocation. With automatic fallback, RECOVER PLUS uses an earlier image copy or LOAD REPLACE LOG YES or REORG LOG YES log point to recover the space.

**NOTE**

Automatic fallback does not occur if you specify any of the following commands and options:

- INCOPY option on a RECOVER command
- RECOVER UNLOADKEYS command
- REBUILD INDEX command with the WORKDDN option

During automatic fallback, RECOVER PLUS attempts to recover the specified spaces until all of the spaces are recovered or determined to be unrecoverable. The maximum number of severe errors allowed is defined by the ERRCONT installation option (see “ERRCONT=10” on page 548). The recovery step ends with a completion code of 0 if RECOVER PLUS successfully recovers, initially or following a successful fallback, all of the spaces specified. If RECOVER PLUS cannot recover all of the spaces, the completion code is 12.

Message BMC40227 indicates that RECOVER PLUS is beginning automatic fallback processing. Automatic fallback uses messages BMC40879 to indicate the number of failures before fallback is attempted. For example, the following messages indicate that recovery failed for one space and RECOVER PLUS is beginning automatic fallback processing to recover the space:

```
BMC40879I 1 OBJECTS WERE NOT RECOVERED DUE TO SEVERE ERRORS
BMC40227I BEGINNING AUTOMATIC FALLBACK
```
Strategies for point-in-time recovery

RECOVER PLUS is often used to return a set of table spaces and their indexes to a prior point in time, which you can accomplish by using the TOLOGPOINT/TORBA or TOCOPY options. RECOVER PLUS allows several innovative ways to speed up and plan for this type of recovery, such as use of the BACKOUT option and use of Instant Snapshot copies.

Using BACKOUT

When you must recover to a point in time, the table spaces and index spaces are normally undamaged and up-to-date, but the spaces contain the results of unwanted updates. For this scenario, RECOVER PLUS provides an additional alternative for point-in-time recovery. Figure 111 on page 427 illustrates copies that are taken on a weekend, archive logs that are created during the week, and a batch job that runs on Friday night, updating a set of table spaces and indexes. If this job runs incorrectly, a point-in-time recovery is needed to a point before the job.

A point-in-time recovery normally uses image copies and archive and active logs to recover the table spaces. Unless you make image copies of the indexes, you must rebuild the indexes. BACKOUT allows RECOVER PLUS to use the active logs only in this case. Tape mounts are completely avoided and only the parts of the index and table spaces that need to be updated are read and written.
Preparing for a BACKOUT recovery

All that is required for a BACKOUT recovery is a point of consistency. You may quiesce your spaces before a batch job and prepare a RECOVER with the BACKOUT option to use in a batch job failure situation. If you want a copy before the batch job to protect against a media failure or for disaster recovery, use COPY PLUS with the Snapshot Copy capability. You may start your batch job after the COPY has obtained a quiesce point for your application group. You can send the copies, when completed, to your disaster storage vault. If the batch job fails, you can use the RECOVER with BACKOUT strategy. The batch job can then be restarted, or normal activity can resume without the batch run.

Executing a BACKOUT recovery

The following sample syntax illustrates a RECOVER with BACKOUT:
When the BACKOUT option is used for point-in-time recovery, no copies are used. The log records from the current point of the log to the TOLOGPOINT requested are used to back out changes. This recovery is often far cheaper in terms of resources than a conventional forward recovery.

When you use the BACKOUT facility for a point-in-time recovery, you can use log records to recover the indexes. Because no copies are needed, you do not need to make copies of indexes to use this facility. For large indexes, this strategy can be much more efficient than rebuilding the indexes from the recovered table space. This facility also allows RECOVER PLUS to read and write only the portions of the table spaces and index spaces that are affected by the log records backed out.

After a BACKOUT recovery of an index, you cannot recover the index to the current point in time. You must perform one of the following actions:

- Execute the BACKOUT with OUTCOPY YES to create a copy to be used in a later recovery from media damage.
- Run COPY PLUS for DB2 after the BACKOUT. You can make a SHRLEVEL CHANGE copy to reduce the impact on your applications.
- Rebuild the index if it becomes damaged before its next image copy.

**When is BACKOUT recovery allowed?**

The following restrictions apply to a recovery that uses the BACKOUT option:

- Your space must be undamaged and up to date to use BACKOUT. You cannot use BACKOUT if a LOAD or REORG occurred at or after the log point that is requested. You can, however, use BACKOUT with a TOLOGPOINT after a LOAD or REORG, effectively recovering to points after such an event even if no image copy was made.

- If a mass delete on a segmented table that is not defined with DATA CAPTURE CHANGES is backed out, RECOVER PLUS will discover that there is data loss if data pages have been reused and will terminate with an error message.

- If a prior point-in-time recovery has occurred, you cannot request a point-in-time recovery between the PIT_RBA and the START_RBA of the earlier recovery.

- Exception statuses such as RECP are not allowed on a space that is subject to BACKOUT. However, if you are restarting the BACKOUT, the RECP status is allowed.

- The BACKOUT option cannot be used with INDEP OUTSPACE, OUTCOPY ONLY, DROPRECOVERY, or OBIDXLAT.
You cannot use RECOVER INDEX or RECOVER INDEXSPACE through a point in time that is created by BACKOUT.

BACKOUT is not valid for

— LOB spaces
— NOT LOGGED spaces
— indexes defined using an expression

**BACKOUT and indexes**

The BACKOUT feature operates on table spaces and indexes. When you recover a table space to a previous point in time, you must also recover all indexes that are defined on that table space to the same point in time or you must rebuild them. This recovery or rebuild is required whether or not BACKOUT is used. Because BACKOUT operates on indexes and does not require an image copy, you can usually use it to avoid expensive rebuild operations, even on indexes that are never image copied.

An exception occurs when you recover an individual partition of a partitioned table space to a previous point in time. In this case, you will need to rebuild any nonpartitioned indexes (NPIs) that are defined on the table space.

**BACKOUT recovery and restart**

You can restart a job that was using BACKOUT, but you must not change the TOLOGPOINT. You must not *rerun* such a job because RECOVER PLUS does not accept the exception statuses, such as RECP, except in *restart*.

**Using Instant Snapshots**

Intelligent storage devices are available that can provide almost instantaneous copies of data sets at a specified point in time by controlling the placement of data on large arrays of disks. Restoration of such copies is also nearly instantaneous, which can significantly reduce recovery time. RECOVER PLUS can use such copies, made by the BMC COPY PLUS utility with the BMC EXTENDED BUFFER MANAGER (XBM) or SNAPSHOT UPGRADE FEATURE (SUF) products, for the forward recovery of table spaces and indexes.
For more information about Instant Snapshots, see the following BMC documentation:

- EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide for details about Instant Snapshots and supported storage devices
- COPY PLUS for DB2 Reference Manual for details about making Instant Snapshots with the COPY PLUS utility

### Registration of Instant Snapshots

Immediate Snapshots are registered in the BMC BMCXCOPY table with an STYPE of V. They are not registered in SYSIBM.SYSCOPY because they are not in the standard format for copies. Standard DB2 utilities cannot use Instant Snapshots.

If a backup copy (LB or RB) is produced and its associated primary copy (LP or RP) is an Instant Snapshot and as such is registered in BMCXCOPY, the backup is also registered in BMCXCOPY, even if it is a standard copy.

### SNAP phase for Instant Snapshot recovery

The SNAP phase is used to restore table spaces and indexes by using Instant Snapshots. The SNAP phase interfaces with XBM. The SNAP phase is followed by a MERGE phase if required (if there are logs to apply, if output copies are requested, or if keys are extracted). For multi-data-set, nonpartitioned spaces where a DSNUM ALL recovery is requested, a single SNAP phase restores all Instant Snapshots, followed by a MERGE phase.

### INCOPY FULL SNAPSHOT

The INCOPY FULL specification includes a SNAPSHOT option so that you can use a non-registered Instant Snapshot as an input copy.

### Executing an Instant Snapshot recovery

To use Instant Snapshots in recovery, you must specify the XBMID installation option or XBMID on the OPTIONS command. XBMID names the subsystem from which the requested Instant Snapshots had been made.
The following sample syntax provides an example of the syntax needed for a recovery that uses an Instant Snapshot:

```sql
OPTIONS XBMID DBXM
RECOVER TABLESPACE PAYROLL.EMPLOYEE
   TABLESPACE PAYROLL.TAXRATES
   TABLESPACE PAYROLL.BENEFITS
   INDEXSPACE PAYROLL.EMPNUMIN
   INDEXSPACE PAYROLL.EMPNAMEIN
   INDEXSPACE PAYROLL.TAXSTSN
   INDEXSPACE PAYROLL.BENEPLIN
```

**Instant Snapshots and migration**

When you use Instant Snapshots for migration and specify the OBIDXLAT option, RECOVER PLUS automatically detects if the OBIDs have not changed and avoids the overhead of reading and writing all of the pages.

**Instant Snapshot restrictions**

The following restrictions apply to recoveries that use Instant Snapshots:

- Instant Snapshot copies cannot be used as a source when the OUTCOPY ONLY option is specified.

- Instant Snapshot copies cannot be created with the OUTCOPY YES or OUTCOPY ONLY options.

- When Instant Snapshots are made with COPY PLUS for use with RECOVER PLUS for a nonpartitioned, multi-data-set space, GROUP YES is required and all of the copies must be Instant Snapshots if RECOVER DSNUM ALL is specified for the space. For a DSNUM n recovery, these restrictions do not apply.

- If an Instant Snapshot copy is used to recover the object, REDEFINE YES has no effect and the data sets are processed as if REDEFINE NO had been specified.

**XBM diagnostics**

To help with performance and problem diagnosis associated with Instant Snapshots, if you have EXTENDED BUFFER MANAGER (XBM) version 5.6 or later and you use Instant Snapshots for recovery, you can print the XBM diagnostics in RECOVER PLUS output. To do so, you include an AFRDBG DD in your RECOVER PLUS JCL, for example:

```sql
//AFRDBG   DD DUMMY
```
Instant Snapshot use by other BMC utilities

Instant snapshots are recognized and used by other BMC products that access the BMCXCOPY table in which these copies are registered.

- COPY PLUS for DB2 produces these copies for recovery.
- RECOVERY MANAGER for DB2 reports these copies and recognizes them for recovery when generating and optimizing the necessary JCL.
- The COPY PLUS MODIFY command deletes the Instant Snapshots from BMCXCOPY and the ICF catalog through its standard functions. All commands operate on the Instant Snapshots registered in BMCXCOPY and their associated standard backup copies. However, template-generated copies are standard copies only, not Instant Snapshot copies.
- UNLOAD PLUS for DB2 unloads data from these copies.

IBM FlashCopy

RECOVER PLUS can use IBM FlashCopy image copies as a recovery resource. The FlashCopy image copies are handled in much the same way as BMC Snapshot copies and are processed by the RECOVER PLUS SNAP phase.

- A consistent FlashCopy is processed in the same way as a BMC Online Consistent Copy. (For more information about Online Consistent Copies, see “Using Online Consistent Copies” on page 75.)

**NOTE**

Note the following considerations:

- RECOVER PLUS supports consistent FlashCopies with log apply when you do not specify INCOPY.
- RECOVER PLUS supports consistent FlashCopies with INCOPY TOCOPY specified.
- RECOVER PLUS does not support consistent FlashCopies when you specify both INCOPY and log apply (RBA or LOGPOINT specified in the INCOPY specification).

- An inconsistent FlashCopy is processed as a SHRLEVEL CHANGE copy in the same manner used for inconsistent BMC Snapshot copies.
- Sequential copies created from a consistent FlashCopy are processed in the same way as sequential BMC Online Consistent Copies.
- Sequential copies created from an inconsistent FlashCopy are processed as SHRLEVEL CHANGE copies.
The OPTION command RESOURCE SELECTION syntax supports FlashCopy (FC). (page 120) FlashCopy is also supported in the installation options LOCCPSEL (page 551) and REMCPSEL (page 551). The default processing order for local copies is FC, LP, LB. The default processing order for remote copies is RP, RB, FC.

The FLASHCOPY option is available to the Non-registered copy or INLOG specification with INCOPY FULL. (page 195)

The DSNAME description of the INCOPY specification accommodates FlashCopy support. (page 204)

---

**Working with ASCII and Unicode data**

ASCII data affects the rebuilding of indexes. The ASCII definition of a table is directly considered in the following tasks:

- padding of variable length character columns in an index key
- assignment of a default value to an index created on a new, unpopulated column defined with NOT NULL WITH DEFAULT
- assignment of a default value to an index created on a new, unpopulated column defined with NOT NULL WITH DEFAULT 'xyz'.

RECOVER PLUS determines the encoding scheme, the proper padding character, and default values to deal with these issues. RECOVER PLUS is fully compatible with ASCII and UNICODE tables.

---

**Managing sort performance**

Performance of a RECOVER PLUS recovery is routinely measured by the elapsed time of the total job. One of the biggest factors that influences elapsed time is sorting log data, index keys, or both. RECOVER PLUS offers many options that enable you to control the efficiency of the sort and customize the sort based on your environment.

Multiple concurrent key sorts provide the greatest performance gain for the product. For each table space, RECOVER PLUS distributes the index keys for all indexes being rebuilt over a user-defined number of sorts and runs these sorts in parallel. For a partitioned table space, if the partitioning index is being rebuilt, RECOVER PLUS does the rebuild of each partition at the completion of the MERGE or UNLOAD for each partition of the table space and this may happen concurrently with the MERGE or UNLOAD for the next partition (if the user-specified number of sorts is not exceeded).
When RECOVER PLUS has input all of the keys to the sort for a group of indexes, RECOVER PLUS then performs the REBUILD operation for each index in a subtask. There could be one subtask for each group of indexes that were sorted in the previous MERGE or UNLOAD phases. So, index REBUILDs can run concurrently.

RECOVER PLUS determines the number of concurrent REBUILDs for each table space during analysis but never exceeds the user-specified number of sorts. Once RECOVER PLUS schedules all of the REBUILDs for a table space in multiple subtasks, it begins the next MERGE or UNLOAD for a different table space if the user-specified number of sorts will not be exceeded.

RECOVER PLUS uses BMC BMCSORT technology for sorts. This technology provides RECOVER PLUS with more control of the sort process than external sort routines provide. This added control helps prevent memory-related problems during the sort process.

RECOVER PLUS allocates the amount of resources to each sort process based on the amount of work that RECOVER PLUS determines that the sort process will perform. RECOVER PLUS also dynamically detects excess available memory and allocates a percentage of them to the sort processes.

The following areas are of major importance to sort performance and can be customized:

- number of parallel sorts allowed
- central and expanded storage
- DASD work space

In general, the more central and expanded storage available to the sort, the better the sort performs. In most cases, the RECOVER PLUS and sort installation default values are sufficient, but if the defaults are not appropriate for your needs, you can use RECOVER PLUS options to override them.

**Recommendations for a large number of concurrent sorts**

RECOVER PLUS automatically checks the available memory in the system (total and in use) and estimates the amount of memory needed to do each sort. If enough memory is available without impacting the system, RECOVER PLUS starts up to MAXKSORT and MAXLSORT sorts concurrently.

If not enough memory is available, RECOVER PLUS automatically reduces the values for MAXKSORT and MAXLSORT to the maximum number of sorts that can run concurrently with the available memory. RECOVER PLUS first reduces MAXLSORT as far as necessary until the value of MAXLSORT is 2. If too many sorts for available memory are still indicated, RECOVER PLUS then reduces MAXKSORT until a manageable number of sorts is reached.
RECOVER PLUS does not reduce MAXLSORT to favor MAXKSORT if no indexes are being rebuilt.

In almost all cases, RECOVER PLUS optimizes the number of sorts automatically.

In the rare case that too many sorts are running and performance is degrading, you can

- reduce the number of sorts by setting MAXKSORT
- control the amount of memory used by each sort using SMCORE

**NOTE**

You can set the SMCORE and MAXKSORT options in the $AFROPTS installation macro or you can code them in the OPTIONS statement in the SYSIN for each RECOVER PLUS job.

---

**Sort file size estimation**

The sort file size for each sort required is the major factor that affects the distribution of workloads and resources between separate sorts.

In almost all cases RECOVER PLUS can automatically estimate the file size using DB2 statistics, data set size, image copy size, or DB2 log ranges.

If the file size estimates that result from the use of the defaults are inaccurate and cause performance to suffer, RECOVER PLUS offers ways to override these estimates with the NUMREC option (page 115, page 224, and page 234).

**Log sort default**

RECOVER PLUS attempts to estimate a file size for a log sort by calculating the number of pages contained in the SYSIBM.SYSLGRNX log range that is required to recover the all the objects in a step and assuming there will be one log record per page input to the sort.

Depending on the number of objects in the recovery, the size of log records, and the number of records for other objects, this strategy may or may not be reasonable. The values that are reported by the LOGSCAN command provide a better estimate for NUMREC and AVGRECSZ.

For more information about sorting the log, see “Understanding the LOGSORT strategy” on page 418.
Key sort default

RECOVER PLUS can determine index sort work space estimates based on RUNSTATS or real-time statistics.

Use of RUNSTATS

For index rebuilds for DB2 Version 8 and earlier or for DB2 Version 9 and later with the DB2 UT SORT DATA SET ALLOCATION field (UTSORTAL subsystem parameter) set to NO (the default value), RECOVER PLUS uses the value that is stored by RUNSTATS in the CARD column of SYSIBM.SYSTABLES to determine the cardinality of the table. If the index is for a partitioned table space, the number of rows is divided equally among all partitions. If a -1 is in the CARD column, RECOVER PLUS (by default) assumes that RUNSTATS has not been run and calculates an estimated file size. This calculation is based on the size of the table space, the maximum rows per page, and the number of indexes.

NOTE

If you chose the NUMREC CALC method and a -1 is found in the CARD column, and the underlying table space for the index is being recovered at the same time, this automatic calculation is not performed. In this case, no file size estimate is made and sort performance could suffer and sort capacity problems could occur or both.

For best performance on index sorts, use NOWORKDDN and keep your catalog statistics up to date with BMCSTATS, RUNSTATS, or the statistics option in COPY PLUS. (If you use WORKDDN, RECOVER PLUS determines the exact number of records to sort, but I/O to the work file may degrade performance.)

Use of real-time statistics

For index rebuilds for DB2 Version 9 and later with the DSN6SPRM UTSORTAL parameter set to YES, RECOVER PLUS uses real-time statistics by using the TOTALROWS value from table SYSIBM.SYSTABLESPACESTATS for single-table table spaces.

NOTE

If UTSORTAL is set to YES and you perform a migration, TOTALROWS may not be accurate. If it is not accurate after the migration, you must reset the value to NULL. However, TOTALROWS is generally accurate and you may not need to reset its value in the following situations:

- if you migrate data daily and the size of the space does not change much
- if you include the real-time statistics tables in your migration
Separate index rebuilds

When rebuilding multiple indexes of varying lengths on the same table space and not enough sorts are allowed to distribute the indexes with different length to separate sorts, sort performance is improved if some indexes are rebuilt in separate jobs because RECOVER PLUS pads all keys to the length of the longest key directed to the same sort to avoid a variable length record sort. Rebuilding some indexes in separate jobs also reduces the amount of sort work space that is required.

RECOVER PLUS sort parameters

RECOVER PLUS has several parameters that are set at installation time that affect sorting.

Setting the SMCORE option

RECOVER PLUS provides the SMCORE installation option to give you control, when necessary, over the amount of memory that BMCSORT uses during a recovery job. SMCORE is specified at installation time and is part of the AFR$OPTS installation options module. You can override SMCORE at run time by using the OPTIONS command. This option contains two parameters, total memory and below-the-line memory.

NOTE

BMC recommends that you specify zero for both SMCORE values (as 0K or 0M) to allow the sort routine to determine the optimal amount of storage to allocate above and below the 16-MB line.

The following values tell RECOVER PLUS to determine the appropriate amount of memory to use for each sort based on the following criteria:

- value that you specified for REGION in either your JCL or system exits
- amount of memory that is available during optimization
- number of sorts to process

— Using KSORTSHARE YES, the maximum number of sorts is:

\[
\text{MAXKSORT} + \text{MAXLSORT}
\]

— Using KSORTSHARE NO, the maximum number of sorts is:

\[
\text{MAXLSORT} + (\text{MAXKSORT} \times \text{MAXLSORT})
\]
For more information, see “Setting the MAXLSORT and KSORTSHARE options” on page 441 and “Setting the MAXKSORT option” on page 439.

**Total memory**

The first parameter value of the SMCORE option tells RECOVER PLUS how much total memory, above and below the 16-MB line, that you want BMCSORT to use during a single invocation. You can enter the value as a number kilobytes (using the suffix K), or megabytes (using the suffix M).

BMC recommends that you specify zero (as 0K or 0M) to allow RECOVER PLUS to determine the optimal amount of storage.

In addition to 0 KB or 0 MB, valid values are 4096 KB through 65536 KB (using the suffix K) or 4 MB through 64 MB (using the suffix M). Regardless of whether RECOVER PLUS determines the value for total memory or you specify a value, RECOVER PLUS multiplies this value by the number of required sort processes to get a value for the total memory required for the current job. Depending on the workload and system environment, RECOVER PLUS distributes this total memory among the sort processes for the job.

For example, if you specify 4096 KB and RECOVER PLUS determines that it needs four sort processes for this job, RECOVER PLUS calculates that it needs 16384 KB total memory for the job. If the workload for each sort process is different, RECOVER PLUS invokes BMCSORT for each sort process with varying amounts of memory. Some of these amounts will be lower and some of these amounts will be higher than the 4096 KB that you specified.

The following additional considerations apply to the SMCORE option:

- RECOVER PLUS always attempts to honor the value that you specified. Improper setting could lead to performance degradation and memory shortage. The region size available for your recovery job with the value that you specify for this subparameter can constrain the number of sort processes that RECOVER PLUS could start. Because the region size must include space for buffers and other required structures, the entire region size is not available for sort processing.

- When you allow RECOVER PLUS to optimize total memory, RECOVER PLUS does not use more than the value of your region parameter and does not use more than some reasonable percent of available memory.

- If you specify an SMCORE value that is larger than your region size or the available memory on the system, RECOVER PLUS will never use more than the value of your region size or the available memory on the system.
Below-the-line memory

The second parameter value of the SMCORE option tells RECOVER PLUS how much memory below the 16-MB line that you want BMCSORT to use during a single invocation. You can enter the value as a number kilobytes (using the suffix K), or megabytes (using the suffix M).

BMC recommends that you specify zero (as 0K or 0M) to allow RECOVER PLUS to determine the optimal amount of below-the-line memory to use. In addition to 0 KB or 0 MB, valid values are 256 KB through 4096 KB (using the suffix K) or 1 MB through 4 MB (using the suffix M).

BMCSORT never needs more than 256 KB of memory below the line. Specifying a value greater than this number can limit the number of sort tasks that RECOVER PLUS can start concurrently.

Setting the MAXKSORT option

RECOVER PLUS determines the optimal number of sorts executing concurrently, depending on available resources. You can specify the maximum number of concurrent sorts with the MAXKSORT option.

Under normal circumstances, allow RECOVER PLUS to control the number of sorts processed concurrently. If system resources are constrained or other problems arise, you can change the MAXKSORT option to limit the number of sorts that are running concurrently.

The MAXKSORT installation option or the MAXKSORT option in the OPTION command statement determines the maximum number of index key sorts that RECOVER PLUS can run in parallel and the maximum number of indexes (or index partitions) that RECOVER PLUS can rebuild in parallel for each table space. Setting the value of MAXKSORT too low could significantly increase the time required to rebuild indexes. Setting the value of this parameter too high could result in overuse of system resources, which could cause total system performance degradation as well as performance degradation of the recovery job.

For indexes on nonpartitioned spaces, the optimum value for MAXKSORT is the largest number of indexes that will be rebuilt for any table space in the recovery job. For example, if a recovery job rebuilds five indexes for table space A and three indexes for table space B, set MAXKSORT to a minimum value of 5. For this example, if MAXKSORT is set to a minimum of 8, RECOVER PLUS might be able to overlap the index rebuilds for table space A with the recovery of table space B, reducing the elapsed time of the job. If the number of indexes to rebuild is so large that you cannot set MAXKSORT to that number, set MAXKSORT as high as possible. RECOVER PLUS groups indexes with similar key lengths together to form an index group.

RECOVER PLUS displays the index group assigned to each index in message BMC40865I in the Object Summary.
RECOVER PLUS sort parameters

For indexes on partitioned spaces, RECOVER PLUS uses the value of MAXKSORT and attempts to divide partitions into groups to allow parallel unloading of keys. The number of partition groups is balanced with the number of indexes to achieve the largest number of parallel sorts and index rebuilds without increasing the total number of bytes sorted or the size of the largest sort.

As with nonpartitioned indexes on nonpartitioned table spaces, nonpartitioned indexes are placed in index groups. The number of parallel sorts is the number of partitions in a partition group multiplied by the number of index groups. For a partitioning index rebuilt with nonpartitioned indexes, the partitioning index may be assigned to its own index group or may be placed in a group with nonpartitioned indexes. The number of indexes that are built in parallel equals the number of index groups.

For example, assume you have a partitioned table space with 30 partitions and four nonpartitioned indexes with key lengths of 200, 120, 60, and 50. If you specify OPTION MAXKSORT 15, RECOVER PLUS creates three index groups with the 200- and 120-byte keys having their own group and the 60- and 50-byte keys sharing a group. Each partition group contains five partitions, and there are six groups of partitions.

The partitions are assigned to groups based on the estimated number of rows in each partition such that, as much as possible, the partition groups contain an equal number of rows. For this example, there are 15 parallel index key sorts (five partitions in each partition group multiplied by three index groups). Three indexes are built in parallel because there are three index groups.

The maximum value that you can assign to MAXKSORT is determined by the amount of below-the-line and above-the-line memory available in the region in which the RECOVER PLUS job runs. RECOVER PLUS does not allow a value for MAXKSORT that is larger than that which can be supported by the available memory. The formula to determine the maximum value allowed for MAXKSORT is as follows:

\[
a = \frac{\text{above-the-line memory}}{30 \text{ MB}} + 1
\]

\[
b = \frac{\text{(below-the-line memory} - 1.5 \text{ MB})}{300 \text{ KB}} + 1
\]

\[
\text{MAXKSORT} = \min(a, b)
\]

For example, in a region with 7.5 MB below the line and 720 MB above the line, the maximum value for MAXKSORT is 21:

\[
a = \frac{720 \text{ MB}}{30 \text{ MB}} + 1
\]

\[
b = \frac{7.5 \text{ MB} - 1.5 \text{ MB}}{300 \text{ KB}} + 1
\]

\[
\text{MAXKSORT} = \min(25, 21)
\]

\[
\text{MAXKSORT} = 21
\]
You should also consider the value of MAXLSORT when setting the value of MAXKSORT. For more information, see “Setting the MAXLSORT and KSORTSHARE options” on page 441.

**Setting the MAXLSORT and KSORTSHARE options**

RECOVER PLUS can run multiple log sorts and parallel MERGE phases. SNAP and RESTORE phases (for inline copies) also run in parallel.

The MAXLSORT option, which you can set in the installation options (page 553) or on the OPTIONS command (page 138), specifies how many log sorts can run in parallel and also determines the number of MERGE/SNAP/RESTORE phases that can run in parallel, regardless of log requirements. In the case of parallel phases, the phases run in subtasks and MAXLSORT sets the number of subtasks.

Use of the MAXLSORT option is dependent on the amount of available memory. BMC suggests that a MAXLSORT value of 10 to 12 is practical. The default value of the MAXLSORT installation option is 0, which allows RECOVER PLUS to determine the number of sorts.

If you specify MAXLSORT 1, you set up your job to run as it would in RECOVER PLUS version 8.1.00 and earlier and turn off parallel log sorts and parallel MERGE/SNAP/RESTORE phases. These phases then run serially in the main task in the order in which they are scheduled by the RECOVER PLUS planning component.

RECOVER PLUS must run some phases serially due to constraints such as

- sharing a DSNUM ALL or cabinet copy
- using copies on stacked tape
- sharing a CHANGE ACCUM file

To honor these constraints, the RECOVER PLUS planning component places the phases into groups. Each phase in such groups runs serially in a subtask. RECOVER PLUS distributes the processing of these groups to balance the workload.

When a table space is recovered and associated indexes are rebuilt in the same step, the MERGE phase extracts keys for the indexes. Since each table space generally uses the maximum number of key sorts specified by MAXKSORT, RECOVER PLUS may not be able to run MERGE phases from different table spaces in parallel because each MERGE requires MAXKSORT key sorts. (For more information about MAXKSORT, see “Setting the MAXKSORT option” on page 439.) In this case, you use the KSORTSHARE option, which you can set in the installation options (page 554) or on the OPTIONS command (page 139), to specify if key sorts are shared among RECOVER PLUS table space recoveries (MERGE phases) running in parallel.
When you specify KSORTSHARE YES, which is the default value, up to MAXKSORT key sorts are shared at execution. Key sorts can only be used by one table space at a time. With KSORTSHARE YES, RECOVER PLUS generates UNLOAD phases to extract keys. However, during execution, if a sufficient number of key sorts is available when the first MERGE phase for a table space is started, the MERGE phase extracts the keys and the UNLOAD phase is skipped.

When you specify KSORTSHARE NO, each MERGE phase has its own set of key sorts and RECOVER PLUS can have up to MAXKSORT * MAXLSORT key sorts active at any given time. Since the number of sorts that can be active in a system is fairly small—usually no more than 30—a value of NO for this option may severely limit the number of recovery operations that RECOVER PLUS can perform in parallel when index rebuilds are also requested.

If the values that you specify for MAXLSORT and MAXKSORT result in too many sorts, RECOVER PLUS reduces the value of MAXLSORT until the number of sorts can be accommodated. This reduction in MAXKSORT and MAXLSORT is only necessary when you specify KSORTSHARE NO.

**Setting the RESINV parameter**

The RECOVER PLUS RESINV parameter is used to specify the amount of virtual storage to be reserved below the 16-MB line for program-invoked sorts. This parameter is specified at installation time. RECOVER PLUS recommends that this parameter always be specified as RESINV 0K, which is the default. This value allows the sort utility’s installation default to be used.

**Using the SORTNUM and SORTDEVT parameters**

The SORTNUM parameter controls the actual number of sort work files that are allocated. The SORTDEVT parameter controls the device type to which the sort work files are allocated. These parameters control where each of the sort work files are allocated when dynamically allocating files. For a summary of information about SORTNUM and SORTDEVT, see Table 15 on page 442.

**Table 15 Parameters for DASD Work Space**

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
<th>Description</th>
<th>OPTIONS parameter?</th>
<th>Installation option?</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORTDEVT</td>
<td>SYSDA</td>
<td>dynamic sort work unit name</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>SORTNUM</td>
<td>0</td>
<td>number of sort work files</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>


**Other sort parameters**

Further control of the sort is available through the RECOVER PLUS OPTIONS command as documented in “SORT specification description” on page 118. These OPTIONS command includes the following options:

- SORTDIAG, which turns diagnostic messages on and off
- SORTDYN, which controls and overrides installation defaults for sort work

---

**NOTE**

BMC strongly recommends that you not change the default value for SORTDYN.

---

**Allocation of sort work space**

You can allocate sort work space by using dynamic allocation, hard-coded JCL sort work files, or both. It is critical that you have the right amount of space allocated to handle all of the data without causing performance degradation or abends. Common problems occur because of inefficient placement of the sort work files or not allocating enough space because bad file size estimate is passed to the sort. In most cases, letting BMCSORT dynamically allocate sort work spaces provides good performance.

---

**NOTE**

To determine the size of sort work data sets, see “Index sort work space requirements” on page 287.

---

When you hard-code sort work files in JCL, use the following guidelines:

- Allocate in cylinders.

- Allocate all sort work files on one device type, preferably the device with the fastest data transfer rate.

- If possible, allocate sort work files on different volsers across different channel paths.

- Do not use VIO data sets for sort work.

- Allocate enough sort work space in primary storage to contain all input data.
When sorting multi-gigabyte files, prefer direct JCL allocation of sort work files to
dynamic allocation because of the large number of resources needed. Depending
on the file size, allocate between four to twelve sort work data sets on separate
devices and, preferably, an equivalent number of channel paths for optimal
performance.

While BMCSORT allows the allocation of up to 32 hard-coded sort work files for any
one sort, hard-coding more than 12 sort work files can cause performance to degrade
because of high overhead. Start by allocating four sort work files, and allocate more
only if you must because of problems in obtaining extents.

Strategies for using copies

This section describes strategies for using image copies and cabinet copies.

Image copy data set contents

An image copy data set that is not a cabinet copy never contains pages from more
than one table space or index space. However, using DSNUM ALL in the image copy
utility (the default) makes a copy for all of the partitions or data sets of a table space.
You may override this option by making copies by using DSNUM integer or by using
the BMC COPY PLUS product DSNUM PART feature and dynamic allocation.

Cabinet copy data set contents

A cabinet copy contains copies from multiple table or index spaces. A cabinet copy
may contain DSNUM ALL copies of copies by data set.

Using copies containing multiple data sets

When you need to recover a partitioned or multi-data-set space, use of a copy
containing all of the data sets has the following ramifications:

- An attempt to recover the data sets in separate jobs will cause contention for this
copy.

- For image copies, the recover utility must read the pages of all previous data sets to
recover a specific data set. For cabinet copies, RECOVER PLUS positions to the
desired data set.
If you always recover all of the data sets or have a very small partitioned space and have no need to run separate concurrent recoveries for data sets, an image copy for all data sets may be appropriate. If you have several small table or index spaces that need to be recovered together, a cabinet copy may be appropriate.

The use of index-controlled partitioning or table-space-controlled partitioning has no ramifications for multi-data-set processing.

For more information about multi-data-set processing for a specific RECOVER PLUS syntax option, see the option description in Chapter 3, “RECOVER PLUS syntax.”

Using data sets stacked on tape

RECOVER PLUS accommodates stacked tape input copies by appropriately ordering the phases of recover and using the data sets in order without dismounting wherever possible. DD statements for the input image copy data sets should *not* be included in the JCL; RECOVER PLUS will dynamically allocate them in the correct order. In contrast, DB2 RECOVER requires DD statements with appropriate volume references and retention and label ordering of table spaces in the statements.

You can create stacked copies with the DB2 COPY utility by ordering copy statements and coding appropriate references in the JCL or by using the dynamic allocation capabilities of the BMC COPY PLUS for DB2 utility. If you use wildcarding with COPY PLUS, stacked data sets for table spaces are ordered by database name, table space name, and data set number or partition.

RECOVER PLUS can also create output stacked image copies. If DD statements are coded in the JCL for the output image copies, they determine the output stacking order. If dynamic allocation is used, RECOVER PLUS orders the output stacked copies to match the order of any input stacked copies. If no input stacked copies are used, RECOVER PLUS follows the COPY PLUS ordering conventions for the output stacked copies.

**NOTE**

To recover a space, RECOVER PLUS requires parallel access to all required copies for a merge operation. RECOVER PLUS does not support recovery requests where two or more required copies, such as a full and an incremental copy, are on the same stacked tape. RECOVER PLUS issues the following message in this case:

BMC40087S REQUIRED INPUT COPIES SHARE THE STACKED TAPE AND CAN NOT BE USED FOR MERGE OPERATION

Ensure that input copies that are required to recover any single space do not share a stacked tape.
If you are recovering table spaces and indexes that were copied with COPY PLUS by using wildcarding and specifying copies for all partitions, the following example illustrates the order of the stacking supported:

```
DB.TS DSNUM ALL
DB.IXP PART 1
  .
  .
DB.IXP PART n
DB.IXN
```

In the example, the following representations are made:

- DB.TS is a partitioned table space.
- DB.IXP is a partitioning index space.
- DB.IXN is a secondary index space.

**NOTE**
If you use stacked image copy tapes and also plan to use R+/CHANGE ACCUM to create change accumulation files, you can use BMC products to coordinate the order for maximum efficiency. R+/CHANGE ACCUM orders log records for multiple table spaces by database name, table space name, and DSNUM. Log records for table spaces and indexes are ordered like the preceding copies made by COPY PLUS.

---

**Using image copies and log records for indexes**

This section describes some of the strategies that use image copies and log records to recover indexes.

**INDEXLOG AUTO and IXSIZE**

RECOVER PLUS, COPY PLUS, and RECOVERY MANAGER have options that automate the creation of index image copies and the use of index image copies in recovery.

With IXSIZE option in COPY PLUS you can specify a threshold at which index copies will be made. IXSIZE is also supported by RECOVERY MANAGER.

By using the INDEXLOG AUTO option that is supported by RECOVER PLUS and RECOVERY MANAGER, you can specify the use of an index copy if one is available for recovery. If recovery from an image copy is not possible, RECOVER PLUS and RECOVERY MANAGER will fallback to rebuild the index.
Making and using copies with COPY PLUS

COPY PLUS and RECOVER PLUS coordinate copies of indexes by registering events in the BMCXCOPY table. Registration of these events in BMCXCOPY is supported for all supported versions of DB2. (For more information, see the examples in Chapter 5, “Examples of RECOVER PLUS jobs.”) You can, for example, recover an entire application by using TOCOPY LASTCOPY if COPY PLUS has copied all of the objects.

NOTE
Copies of indexes with the COPY YES attribute are registered in the SYSIBM.SYSCOPY table. RECOVER PLUS can also use these copies for recovery.

Using log records for indexes

RECOVER PLUS can use index log records for recovery. For media failure recovery, recovery after a failed LOAD or REORG, or disaster recovery situations, it is necessary to make copies of the index spaces by using COPY PLUS, the DB2 COPY utility, DSN1COPY, or pack dumps. If you make copies with COPY PLUS or the DB2 COPY utility, they can be used automatically with RECOVER PLUS and log records rolled forward from the registration point. DSN1COPY and pack dump copies must be restored to the space and RECOVER PLUS LOGONLY used to apply log records.

For point-in-time recoveries where the spaces are intact and no LOAD or REORG occurred after the point in time, you can use the RECOVER PLUS BACKOUT feature to reverse index log records to the point in time.
Using R+/CHANGE ACCUM with RECOVER PLUS

The following information pertains to the BMC R+/CHANGE ACCUM for DB2 product. This product is designed to work with RECOVER PLUS to streamline normal and disaster recovery processes. R+/CHANGE ACCUM accumulates log record data for a specified object or group of objects. Log record data for specified objects is extracted and stored in a change accumulation file. For detailed information about using this product, see the R+/CHANGE ACCUM for DB2 User Guide.

Using change accumulation files for multiple objects

Using R+/CHANGE ACCUM, you can select several specific table spaces and define them as a group. When you run a R+/CHANGE ACCUM batch job on the group, you create a change accumulation file of log record data for all of the table spaces in the group. You may elect to include log record data for all of the indexes on all of the table spaces in the group.

You may want to create change accumulation files for multiple objects if you are writing change accumulation files to tape or want to avoid creating too many disk data sets. Objects defined as a group are ordered alphabetically by database name and then table space name. Data sets or partitions are ordered within the same space.

When you need to recover an object that shares a change accumulation data set with other objects, consider the following ramifications:

- An attempt to recover the objects on a file in separate jobs will cause contention for this change accumulation file.

- RECOVER PLUS must read the log records of all previous objects (sequentially) to recover a specific object.

If you always recover all of the objects or do not anticipate many log records between copies and have no need for separate recovers for the objects, a change accumulation file for multiple objects may be entirely appropriate.

You should attempt to coordinate stacked tape copies and change accumulation containing multiple objects so that, if you use them, you combine objects in a similar order. Target table spaces (table spaces targeted for change accumulation) are ordered by the following items:

- database name
- table space name
- data set number
Indexes for each table space follow the table space in order by partitioned indexes first, and then nonpartitioned indexes, in order by database name and index space name. You should use this order when stacking image copies on tape.

The BMC COPY PLUS for DB2 product orders objects in a similar fashion when using wildcards and dynamic stacking. RECOVER PLUS optimizes access for shared resources, but cannot create an optimal plan if both stacked copies and change accumulation files for multiple objects are used and the stacking is not alphabetical.

Using separate change accumulation files for separate data sets

If you generate separate change accumulation files for separate data sets of a table space, you should use caution in also combining other objects in the same change accumulation file. Some situations could cause RECOVER PLUS to create a less than optimal plan for recovering objects.

If a particular data set of a table space shares a change accumulation file with other objects and RECOVER PLUS attempts to use a copy including all data sets or to unload keys for a nonpartitioned index on a partitioned space, the plan may use more tape drives or have to reread parts of a copy or change accumulation file to process.

Planning recovery resources

Build a plan for grouping copies on tape, grouping objects on change accumulation files, and using copies for all data sets of an object that allows RECOVER PLUS to use all resources effectively. RECOVER PLUS will attempt to

- use shared resources in order (copies including all data sets in data set number order; input change accumulation files in order by database name, table space name, and data set number; input stacked copies by file sequence number)

- group use of shared resource without having intervening steps that use other resources

RECOVER PLUS must process all data sets of a partitioned table space together when a nonpartitioned index is being rebuilt or when multiple partitions are unloaded.

When change accumulation files are being created, the batch process must process the objects in the grouping order: database name, table space name, and data set number. Similarly, RECOVER PLUS cannot alter a stacking order for output copies that is given by your JCL.
Tips on grouping recovery resources

This section provides tips on grouping recovery resources.

Combining copies for multiple objects with change accumulation for groups

If you are making a copy of all data sets of a space and also making change accumulation files for the same space, make the change accumulation file for all data sets in the space.

If you are stacking copies and building change accumulation files for groups of objects, you should coordinate the objects in the change accumulation group with the stacked copies. Include all of the stacked objects or a subset in the change accumulation group. Stack the copies in alphabetical order by database name and table space name with indexes included as indicated in “Using change accumulation files for multiple objects” on page 448.

Handle large partitioned objects with separate change accumulation files and copies

If you have very large table spaces that are partitioned, make specific data set (partition) copies and do not stack them with other partitions or other table spaces. Create a change accumulation group for each partition if the change accumulation file will be tape or if a large numbers of updates will occur between image copies.

If the change accumulation file will be disk, and there are not a large number of updates between copies, it may be appropriate to have all of the partitions grouped into the same change accumulation group. This plan allows a straightforward recovery of any partition or concurrent jobs on the partitions.
Reading multiple log files concurrently

RECOVER PLUS can reduce recovery elapsed time by using multiple log readers with the MAXLOGS option.

Rules for using the MAXLOGS option to indicate the number of log readers are as follows:

- **memory usage**
  
  Memory usage will increase by approximately 1.8 MB for each concurrent log reader execution to a maximum of 20MB. For example, if a job has specified MAXLOGS 4, an additional 7.2 MB of storage will be required during the LOG INPUT phase in a RECOVER PLUS job.

- **setting MAXLOGS and MAXDRIVES**
  
  Use the following algorithm to calculate a first try at values for MAXLOGS and MAXDRIVES if you have archive logs on tape (not including logs migrated to tape with DFHSM):

  1. Divide the number of tape drives that you have available for recoveries by the number of concurrent RECOVER PLUS jobs that you plan to run. This number is your value for MAXDRIVES. Do not reduce MAXDRIVES below the maximum number of image copy and change accumulation tapes (input and output) for one object.

  2. Add the higher of 4 or the number of DASD log files that you anticipate reading to MAXDRIVES to calculate your value for MAXLOGS.

  If all your log files are on DASD or are migrated by using DFHSM or a similar product, use values of 0 (zero) for MAXDRIVES and 6 for MAXLOGS. MAXDRIVES 0 leaves the number of tape drives unconstrained.
Allocating output image copy data sets dynamically

RECOVER PLUS uses the OUTPUT command for dynamic allocation of output image copy data sets. The dynamic allocation feature allows you to make image copies of spaces without including DD statements in the JCL. Instead of using DD statements (each of which provides a physical description of only a single data set) you can use the directives (output descriptors) to provide a logical view of how copy data sets are to be created. The effects of allocating copy data sets dynamically with RECOVER PLUS follow:

- eliminate large, complex DD statements
- greatly simplify tape stacking
- restart a failed job automatically (no JCL or other statements to change)
- release unused space when a copy data set is closed
- use symbolic variables and generation data groups (GDGs) to assist in data set name generation

**NOTE**

Dynamic allocation of output copies is required for spaces with more than 254 partitions.

Because an output descriptor is not directly associated with a particular data set, you can use an output descriptor to describe multiple copy data sets. (For JCL examples, see Chapter 5, "Examples of RECOVER PLUS jobs.")

Using copy data set output descriptors

An output descriptor describes the general characteristics of the copy data set, whether it is a disk data set or a tape data set. These characteristics include

- disk or tape unit name
- operating system cataloging requirements for the data set
- model data control block (DCB)
- generic data set name
- largest number of volumes expected to be used
- SMS class information
- for disk data sets:
  - disk space information
  - lists of volumes
for tape data sets:
— stacked tape indicator
— optional data compression
— data set retention period
— data set expiration date

An output descriptor specifies a disk data set or a tape data set. It cannot specify both.

When you want to use dynamic allocation, use an OUTPUT command statement in your SYSIN data set to specify allocation parameters. The OUTPUT command statement must precede your RECOVER command statement as in the following example:

```
OPTIONS INDEXLOG YES ANALYZE YES OUTCOPY ASCODED

OUTPUT SYSCOP1 UNIT CART STACK YES
   DSNAME RDAMS.M IC&TYPE.&DB.&TS.P&DSNUM(+1)
OUTPUT SYSCOP2 UNIT CART STACK YES
   DSNAME AFR.IC&TYPE.&DB.&TS.P&DSNUM(+1)

RECOVER TABLESPACE AFRDB01.TS01 OUTCOPY ONLY
   OUTCOPYDDN(SYSCOP1,SYSCOP2)
INDEXSPACE AFRDB01.IX01 OUTCOPY ONLY
   OUTCOPYDDN(SYSCOP1,SYSCOP2)
INDEXSPACE AFRDB01.IX02 DSNUM 1 OUTCOPY ONLY
   OUTCOPYDDN(SYSCOP1)
TOLOGPOINT LASTQUIESCE
```

“GDGs and symbolic variables in data set name construction” on page 454 provides more information.

The preceding example allocates all image copies based on the output descriptors for SYSCOP1 and SYSCOP2. Data set names are as follows:

```
RDAMS.M.ICLP.AFRDB01.TS01.P00(+1)
AFR.ICLB.AFRDB01.TS01.P00(+1)

RDAMS.M.ICLP.AFRDB01.IX01.P00(+1)
AFR.ICLB.AFRDB01.IX01.P00(+1)

RDAMS.M.ICLP.AFRDB01.IX02.P01(+1)
```

If the copies are being made by partition, you do not need to specify a DD name prefix, as you do with JCL-allocated copies. All partitions of the output copy can be allocated with the same OUTPUT descriptor.

For more information, see the syntax description in “OUTPUT” on page 145.
GDGs and symbolic variables in data set name construction

You can use generation data set groups (GDGs) and symbolic variables to simplify the task of data set name construction when you use dynamic allocation with RECOVER PLUS. You can use a GDG and symbolic variables together in a data set name.

Using GDGs

The GDG format that you use in data set name construction is the same as the format that you use in JCL when you use DD statements to allocate your copy data sets.

When dynamic allocation is used, RECOVER PLUS also provides the option of specifying an input data set, AFRGDG, to provide control cards to be used to define the GDG base if it does not already exist. This data set must contain the control cards to perform an IDCAMS DEFINE, and the symbolic variable, &BASE, which RECOVER PLUS replaces with the GDG base name. For example:

```
DEFINE GDG(NAME(&BASE) LIMIT(7) -
NOEMPTY NOSCRATCH)
```

Using symbolic variables

You can also use symbolic variables when you specify a data set name in an output descriptor. You can represent the variable elements shown in Table 16 on page 455 by using symbolic variables.

You can specify any or all nodes of a data set name by using symbolic variables as in the following example:

```
DSNAME &UID.&TS.&TYPE
```

This example generates data set names containing the ID of the user making the copies, the space being copied, and the type of copy. Another example follows:

```
DSNAME NEWYEAR.&DB.&TS
```

This example combines a real node name with symbolic variables to generate a data set name.

Symbols for numeric variables (&DATE, &TIME, &JDATE, &YEAR, &MONTH, &DAY, &JDAY, &HOUR, &MINUTE, &SECOND, &SEQ, &DSNUM, &LDSNUM, &PART, and &LPART) must be prefixed by a Latin alphabetic character. In the following example, the first statement causes errors, while the second is correct:
Although you can prefix a symbolic variable with an alphabetic character, you cannot append characters. For example, XX&TS is valid, but &TSXX is invalid. &TS.XX is valid.

You can use symbolic variables with GDGs simply by appending the generation number in parentheses in the usual way. For example, &TS(+1).

Table 16  Symbolic variables for specifying data set names (part 1 of 2)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Symbol</th>
<th>Length of result&lt;sup&gt;a,b&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOB name used in the JCL</td>
<td>&amp;JOBNAME</td>
<td>8 bytes max</td>
</tr>
<tr>
<td>STEP name used in the JCL&lt;sup&gt;c&lt;/sup&gt;</td>
<td>&amp;STEPNAME</td>
<td>8 bytes max</td>
</tr>
<tr>
<td>database containing the space being copied</td>
<td>&amp;DB</td>
<td>8 bytes max</td>
</tr>
<tr>
<td>table space or index space being copied&lt;sup&gt;d&lt;/sup&gt;</td>
<td>&amp;TS</td>
<td>8 bytes max</td>
</tr>
<tr>
<td>space name of table or index being copied</td>
<td>&amp;SP</td>
<td>8 bytes max</td>
</tr>
<tr>
<td>data set or partition being copied</td>
<td>&amp;DSNUM, &amp;PART&lt;sup&gt;e&lt;/sup&gt;</td>
<td>2 bytes for 0 - 99, 3 bytes for 100 - 999, 4 bytes for 1000 - 4096</td>
</tr>
<tr>
<td>data set or partition being copied</td>
<td>&amp;LDSNUM, &amp;LPART&lt;sup&gt;e&lt;/sup&gt;</td>
<td>3 bytes for 000–999, 4 bytes for 1000–4096</td>
</tr>
<tr>
<td>job or TSO user ID</td>
<td>&amp;USERID or &amp;UID</td>
<td>7 bytes max</td>
</tr>
<tr>
<td>DB2 subsystem ID</td>
<td>&amp;SSID</td>
<td>4 bytes</td>
</tr>
<tr>
<td>DB2 group attachment name or subsystem ID</td>
<td>&amp;ATTACH&lt;sup&gt;f&lt;/sup&gt;</td>
<td>4 bytes</td>
</tr>
<tr>
<td>current date (in the form YYMMDD)</td>
<td>&amp;DATE&lt;sup&gt;e,g&lt;/sup&gt;</td>
<td>6 bytes</td>
</tr>
<tr>
<td>current time (in the form HHMMSS)</td>
<td>&amp;TIME&lt;sup&gt;e,g&lt;/sup&gt;</td>
<td>6 bytes</td>
</tr>
<tr>
<td>current Julian date (in the form YYDDD)</td>
<td>&amp;JDATE&lt;sup&gt;e,g&lt;/sup&gt;</td>
<td>5 bytes</td>
</tr>
<tr>
<td>current year (in the form YY)</td>
<td>&amp;YEAR&lt;sup&gt;e,g&lt;/sup&gt;</td>
<td>2 bytes</td>
</tr>
<tr>
<td>current month (in the form MM)</td>
<td>&amp;MONTH&lt;sup&gt;e,g&lt;/sup&gt;</td>
<td>2 bytes</td>
</tr>
<tr>
<td>current day (in the form DD)</td>
<td>&amp;DAY&lt;sup&gt;e,g&lt;/sup&gt;</td>
<td>2 bytes</td>
</tr>
<tr>
<td>current Julian day (in the form DDD)</td>
<td>&amp;JDAY&lt;sup&gt;e,g&lt;/sup&gt;</td>
<td>3 bytes</td>
</tr>
<tr>
<td>current hour (in the form HH)</td>
<td>&amp;HOUR&lt;sup&gt;e,g&lt;/sup&gt;</td>
<td>2 bytes</td>
</tr>
<tr>
<td>current minute (in the form MM)</td>
<td>&amp;MINUTE&lt;sup&gt;e,g&lt;/sup&gt;</td>
<td>2 bytes</td>
</tr>
<tr>
<td>current second (in the form SS)</td>
<td>&amp;SECOND&lt;sup&gt;e,g&lt;/sup&gt;</td>
<td>2 bytes</td>
</tr>
<tr>
<td>type of image copy:</td>
<td>&amp;ICTYPE</td>
<td>1 byte</td>
</tr>
<tr>
<td>— F for FULL YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>— I for FULL NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>— A for FULL AUTO or CHANGELIMIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>— D for FULL DSN1COPY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>utility ID</td>
<td>&amp;UTIL</td>
<td>8 bytes maximum&lt;sup&gt;h&lt;/sup&gt;</td>
</tr>
</tbody>
</table>
When you dynamically allocate copy data sets, you can use the STACK YES option to have all copy data sets allocated with the same tape OUTPUT descriptor stacked on tape. The stacking order is determined by the order of the input image copies, if any. If no input stacked copies exist, RECOVER PLUS uses the default stacking order that is used by COPY PLUS wildcard processing. If there is a mix, the copies with corresponding stacked input image copies are stacked first, followed by the remaining copies in default order.

When you dynamically allocate copy data sets, you can optionally stack output copies of the same type (LP, LB, RP, or RB) contiguously on the same set of tapes.

**WARNING**

If you are using TMM (Tape Mount Management), be aware that TMM intercepts any data set allocation whether dynamic or otherwise. If you want the copies on tape and use STACK YES with TMM, add the RECOVER PLUS program AFRMAIN to the TMM exclusion list.
When you want to stack copies of more than one type on tape (such as local site primary and local site backup), you must use a different output descriptor for each type and stack the copy types on different tape units. For example, the following command statements stack local site primary copies of table spaces A.B, C.D, and E.F contiguously on tape unit CARTLP and the corresponding local site backup copies contiguously on tape unit CARTLB:

```
OUTPUT LPCOPY UNIT CARTLP ... options ... STACK YES
... more options
OUTPUT LBCOPY UNIT CARTLB ... options ... STACK YES
... more options
RECOVER TABLESPACE A.B ..... options
OUTCOPYDDN(LPCOPY,LBCOPY) ..... more options
RECOVER TABLESPACE C.D ..... options
OUTCOPYDDN(LPCOPY,LBCOPY) ..... more options
RECOVER TABLESPACE E.F ..... options
OUTCOPYDDN(LPCOPY,LBCOPY) ..... more options
```

**Using RECOVER PLUS with data sharing**

Before you convert from a non-data-sharing environment to a data sharing environment, you should consider the following points:

- **When you are specifying a recovery, you can use TORBA or TOLOGPOINT keywords interchangeably.**

  If you are specifying a point in time before the conversion to a data sharing environment, you should specify an RBA value. If you are specifying a point in time after the conversion to a data sharing environment, you should specify an LRSN value.

- **If BSDS passwords are used in a data sharing environment, RECOVER PLUS will work only if all of the passwords for the group are the same.**

  OPNDB2ID will work under data sharing only if all Resource Access Control Facility (RACF) IDs for the members of the group are the same. The BSDS authorizations must also be the same.

- **RECOVER PLUS supports recovery in a data sharing environment by using log records written before data sharing was enabled. You can recover table spaces by using image copies that were made before data sharing was enabled.**

- **The DB2 RECOVER utility does not support recovery after data sharing is disabled if the recovery requires log records that were created while data sharing was active (For more information, see DB2 for z/OS Data Sharing: Planning and Administration).**
RECOVER PLUS does support such recovery requests. However, if you re-enable data sharing and any table spaces or indexes have not been copied since before data sharing was originally enabled, BMC recommends that you copy them. Failure to do so may render those objects unrecoverable.

Forward recovery of index spaces after altering a VARCHAR column

RECOVER PLUS supports the forward recovery of an index that has had a VARCHAR column altered. RECOVER PLUS needs information from the header page and the directory page to recover such an index space.

- The header page is always the very first page of any partition or the very first piece of a nonpartitioned index (NPI).
- In general, a directory page is page 4 for any partition or NPI, but the directory page could be any page greater than page 4 if the partition or NPI has been migrated and has not been rebuilt after an ALTER VARCHAR update. BMC strongly recommends that you rebuild or REORG such indexes.

If a multi-data-set, nonpartitioned index contains a VARCHAR column that has been altered, the very first piece of the NPI must be included in any request to recover the NPI. RECOVER PLUS must have at least a header page to process the recovery correctly. If the NPI was altered prior to migrating, and the index was not rebuilt or reorganized, the recovery will fail with the following error if the directory page is contained in a piece that is not included in the recovery.

BMC40855 CANNOT OBTAIN DIRECTORY PAGE INFORMATION. REQUESTED RECOVERY IS NOT POSSIBLE.

In the case of an ALTER VARCHAR, RECOVER PLUS reorders the recovery request to use the correct order of DSNUM. For example, RECOVER PLUS accepts the following command statements, but reorders the request to process DSNUM 1 first:

RECOVER INDEX (I1) DSNUM 5
RECOVER INDEX (I1) DSNUM 1
Supporting real-time statistics in RECOVER PLUS

RECOVER PLUS enables real-time statistics as follows:

- RECOVER PLUS resets the COPY-related counters, even during a recovery to current, if it is making an output image copy during a recovery.
- RECOVER PLUS does not update any real-time statistics during an OUTCOPY ONLY execution.

Table 17 provides the columns in SYSIBM.SYSTABLESPACESTATS\(^1\) and SYSIBM.SYSINDEXSPACESTATS\(^1\) that the RECOVER PLUS RECOVER command updates.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Updated value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPDATESTATSTIME</td>
<td>timestamp of update</td>
</tr>
<tr>
<td>COPYLASTTIME(^a)</td>
<td>timestamp</td>
</tr>
<tr>
<td>COPYUPDATEDPAGES(^a)</td>
<td>0</td>
</tr>
<tr>
<td>COPYCHANGES(^a)</td>
<td>0</td>
</tr>
<tr>
<td>COPYUPDATELRSN(^a)</td>
<td>null</td>
</tr>
<tr>
<td>COPYUPDATETIME(^a)</td>
<td>null</td>
</tr>
</tbody>
</table>

\(^a\) This field is updated by the RECOVER PLUS RECOVER command only when an output image copy is made during a recovery. Because making an output image copy during recovery is not an option with the DB2 RECOVER utility, the DB2 RECOVER utility never updates these fields.

The DB2 REBUILD utility updates some reorganization-related statistics in SYSIBM.INDEXSPACESTATS\(^1\), as well as the REBUILDLASTTIME column. Table 18 on page 460 summarizes the support for updating these statistics when you use the RECOVER PLUS REBUILD command.

---

1. In versions earlier than DB2 Version 9, these tables are SYSIBM.INDEXSPACESTATS and SYSIBM.TABLESPACESTATS.
Recovering altered spaces

RECOVER PLUS supports online schema evolution including the recovery of spaces that have had partitions added or rotated, or that have had their limit keys changed.

Added partitions

If you add a partition with an ALTER statement or if a partition is automatically added with a partition-by-growth universal space, you can perform the following recoveries with RECOVER PLUS:

- You can recover the partition to the current time.
- You can recover the partition to a point in time after the partition was added.
- You can recover spaces to a point in time before the partition existed (prior to the ALTER ADD PARTITION), and RECOVER PLUS creates the partition with no data.

If a point-in-time recovery causes a partition-by-growth table space to shrink and leave the space empty, the underlying data set will contain a header and space map. The number of partitions will not be decreased in the catalog.

**NOTE**

For universal table spaces, if you use the LASTQUIESCE option and the quiesce is prior to the ALTER ADD PART, the recovery will fail. You must specify a hard-coded RBA. If the partition was added for a partition-by-growth space, you can use the LASTQUIESCE option.

---

**Table 18  Columns updated by the RECOVER PLUS REBUILD command for real-time statistics**

<table>
<thead>
<tr>
<th>Column name</th>
<th>Updated value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPDATESTATETIME</td>
<td>timestamp of update</td>
</tr>
<tr>
<td>REBUILDLASTTIME</td>
<td>timestamp</td>
</tr>
<tr>
<td>REORGINSERTS</td>
<td>0</td>
</tr>
<tr>
<td>REORGDDELETES</td>
<td>0</td>
</tr>
<tr>
<td>REORGAPPENDINSERT</td>
<td>0</td>
</tr>
<tr>
<td>REORGPSEUDODELETES</td>
<td>0</td>
</tr>
<tr>
<td>REORGMASSDELETE</td>
<td>0</td>
</tr>
<tr>
<td>REORGLEAFNEAR</td>
<td>0</td>
</tr>
<tr>
<td>REORGLEAFFAR</td>
<td>0</td>
</tr>
<tr>
<td>REORGNUMLEVELS</td>
<td>0</td>
</tr>
</tbody>
</table>
In DB2 Version 9, when you are performing a drop recovery or a migration, the source definition must match the target definition, including the number of partitions. For partition-by-growth (PBG) universal spaces, if the number of partitions for the source has grown to more than one partition, the source would no longer match the target, and the drop recovery or migration fails.

**NOTE**
For DB2 Version 9, a possible solution for the failure is to load the target table with enough data so that the number of data sets match the source. Then perform the migration or DROPRECOVERY. In DB2 Version 10, NUMPARTS and MAXPARTITIONS are compatible. Use NUMPARTS when defining the PBG target so the number of target data sets match the source.

---

**Rotated partitions**

If you rotate (ALTER ADD ROTATE) a partition, you can perform the following recoveries with RECOVER PLUS:

- You can recover the partition to the current time.
- You can recover the partition to a point in time after the rotate.

You *cannot* recover table or index space partition to a point in time before the partition was rotated.

**Changed limit keys**

If you change the limit keys on a partitioned space, the affected partitions are placed in REORP status and you cannot use the affected partitions until you run a REORG on those partitions. DB2 Version 8 added the REBALANCE feature to the REORG utility that changes limit keys and does a REORG of the specified partitions in one step, which avoids having the spaces put in REORP status.

If you recover to a point in time prior to the change, the spaces will need to be REORGed, and will be placed in REORP status.
Recovering encrypted copies

The use of encryption protects sensitive company information and prevents security failures. RECOVER PLUS support for encryption allows you to recover using image copies that are protected from unauthorized access to the sensitive information. For the recovery, RECOVER PLUS uses full and incremental encrypted copies made by COPY PLUS. For information about making encrypted copies, see the *COPY PLUS for DB2 Reference Manual*.

**NOTE**

Encryption is a feature of the Recovery Management for DB2 solution and requires a valid Recovery Management solution password.

Encryption is based on standard secret key encryption algorithms. In COPY PLUS, you select encryption based on one of three following standard algorithms:

- the ANSI Data Encryption Algorithm (DEA)\(^2\) with a 64-bit key
  
  This is the default algorithm. This algorithm is also known as the U.S. National Institute of Science and Technology Data Encryption Standard (DES).

- the Triple Data Encryption Standard (TDES) with a 128-bit key

- the Advanced Encryption Standard (AES)\(^3\) with a 128-bit key

COPY PLUS supports encryption of plaintext image copies or decryption of cipher text image copies. *Plaintext or clear text is data in normal, readable form. (COPY PLUS standard image copies are plaintext.) Encrypted text or cipher text is data that has been converted to mask its meaning from an unauthorized recipient.*

### Requirements for encryption

To specify that you want to recover by using encrypted copies made by COPY PLUS, you must:

- run RECOVER PLUS on a processor that supports encryption

---


Key data set

Support for encryption in RECOVER PLUS relies on a user-created and maintained data set, called the key data set. The key data set contains essential encryption key information. COPY PLUS requires the key data set to make encrypted copies. RECOVER PLUS requires the same or an identical key data set used by COPY PLUS to recover by using encrypted copies.

Key data set requirements

You must perform the following tasks for the key data set:

- Create the key data set.

RECOVER PLUS requires that the key data set be a fixed or fixed block physical sequential data set with a logical record size (LRECL) of 80. RECOVER PLUS requirements for the contents of the data set are specified in “Key data set contents” on page 465. Any variation from these requirements could prevent RECOVER PLUS from recovering the encrypted image copy.

**NOTE**

BMC recommends that you use the same key data set that COPY PLUS used to encrypt the image copy or a duplicate of that COPY PLUS key data set. A duplicate data set is required if you are migrating data from one system to another, and the key data set is not shared between the systems or you are recovering data at a remote site.
Identify the key data set to RECOVER PLUS.

The KEYDSNAM installation option (page 552) specifies the key data set name. After you specify the key data set name, RECOVER PLUS dynamically allocates the data set when it is needed.

If RECOVER PLUS attempts to recover an encrypted image copy and you have not specified the key data set name in the installation options, RECOVER PLUS issues the following message:

```
BMC40020I ENCRYPTION KEY DATA SET NOT SPECIFIED IN OPTIONS MODULE
```

If RECOVER PLUS attempts to recover an encrypted image copy and the key data set is not cataloged, RECOVER PLUS issues the following message:

```
BMC40020I ENCRYPTION KEY DATA SET NOT CATALOGED
```

Maintain the key data set.

Periodically, you may want to change encryption keys. You cannot edit the key data set while any utility that is using the key data set is inflight. You need to schedule time to maintain the data set. You must take care when you maintain the data set because incorrect entries in the data set might prevent COPY PLUS from encrypting your image copies or prevent RECOVER PLUS from recovering from a previously encrypted image copy.

Provide appropriate security for the key data set to protect it from unauthorized access.

**NOTE**

If a non-registered encrypted image copy is used in the INCOPY clause and the registration timestamp is not known, a timestamp with a value greater than the timestamp of the desired encryption key, but less than subsequent encryption keys can be used in place of the registration timestamp. Due to security, the administrator of the key data set will need to determine this value.

Maintain backups of your key data set either with DFSMSHsm or some other facility.
Key data set contents

The key data set contains one or more rows of 80 characters per row. RECOVER PLUS ignores any characters in columns 72 through 80. Each row contains

- one encryption key
- a corresponding timestamp
- an optional encryption algorithm identifier
- an optional comment

These fields are separated by one or more blank characters. The first character of the comment is an asterisk. Rows are ordered in the data set by timestamp with the most recent timestamp first. The current key is the key in the first row. The format of the key data set row is:

<table>
<thead>
<tr>
<th>Key value</th>
<th>Timestamp</th>
<th>Encryption algorithm ID</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0ABCDEF123456789FEDCBA000111111' 2009-11-23-12-00</td>
<td>*128 bit DES encryption</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'123456789ABCDEF1'              2009-08-23-11-10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'723DE6789000DEF1'              2008-12-12-16-40</td>
<td>DES 64 bit DES encryption</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'723DE6789000DEF123DE6789000DEF1' 2008-12-12-14-00</td>
<td>AES *128 bit AES encrypt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'F1F2F3F4F5F6F7F8'              2008-01-01-12-00</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

RECOVER PLUS uses the contents of the key data set to determine a key value for decryption of image copies. RECOVER PLUS uses either the first key whose timestamp is less than the timestamp in BMCXCOPY or the timestamp specified in the INCOPY specification.

Encrypted image copies are registered in BMCXCOPY. As with SYSCOPY registration, BMCXCOPY registration includes a timestamp specifying when the copy was registered. RECOVER PLUS uses this timestamp to find the correct key value in the key data set. For more information about the registration of encrypted copies, see “Registration for plaintext image copies” on page 468.

For example, if RECOVER PLUS selected an image copy for a recovery from BMCXCOPY with a timestamp of 2009-02-12-10.00, the encryption key and DES algorithm in the third row in the example key data set above is selected.
Key value

RECOVER PLUS supports both 64-bit and 128-bit keys. (See “Encryption algorithm identifier”.) The key data set can contain either or both key sizes. The key value is a clear key represented in the key data set as a string of 16 or 32 hexadecimal digits in the following format:

X’dd...

The X and the quotes are required. The X must occur in the first column and be upper case.

Timestamp

The date, hour, and minute string uses following formats:

yyyy-mm-dd-hh-mm

or

yyyy-mm-dd-hh.mm

The values are decimal numbers and are padded on the left with a zero if necessary. The timestamp must be separated from the key value by at least one blank space.

Encryption algorithm identifier

An encryption algorithm identifier is optional in the key data set. The encryption algorithm identifiers supported are

- DES for Data Encryption Standard (for 64-bit keys)
- DES for Triple Data Encryption Standard (for 128-bit keys)
- AES for Advanced Encryption Standard (requires 128-bit keys)

The encryption algorithm identifier defaults to DES if no identifier is provided. If you provide an identifier, you must separate it from the timestamp by at least one blank. RECOVER PLUS distinguishes between the two varieties of DES based on the length of the key (64-bit or 128-bit).

Comments

Comments are optional in the key data set. A comment begins with an asterisk that is separated from the preceding field by at least one blank.
Key data set management

The security of the encrypted COPY PLUS image copies and the ability of authorized individuals to use RECOVER PLUS to recover DB2 spaces using these image copies depends on the careful management of the key data set. BMC recommends that you develop a simple and well-documented mechanism to manage key data sets.

BMC recommends that you maintain one key data set shared by all systems with access to the data set. Multiple distinct key data sets create difficulty with key data set management because you must ensure that the key data set that is used to encrypt an image copy is also used for recovery with that encrypted image copy.

Consider all of the following items as you manage your key data set:

- Protect the key data set on the local system and duplicates on remote systems against unauthorized access.

  Most attempts to access encrypted data occur as unauthorized access to the key data set. You should protect the key data set against unauthorized access during shipping with either a secret key or public key encryption. If key data set is not encrypted during shipping, it should never be shipped under the same cover as the encrypted image copies.

- If you plan to use encrypted image copies at your disaster recovery site, be sure that the processor at the site supports encryption.

  Remote disaster recovery sites may require a duplicate key data set for recovery purposes.

- Because the timestamps that are used for recovery are taken from the BMCXCOPY table, a change in time zones between the site where COPY PLUS made the encrypted image copies and the disaster recovery site will not affect recovery.

  The possibility exists, however, that a time zone change might invalidate a key data set for creating image copies at the remote site. If this is the case, you will need a new key data set with local times for generating encrypted image copies at the remote site.

- Limit updating of the key data set to authorized individuals.

  Generating a new current key by inserting a new first row in the key data set limits the amount of data exposed if the current key is compromised. Do not modify existing rows in the key data set because image copies may exist that will require the keys for recovery. It is important that duplicate key data sets on remote systems also contain this new row, and that backups of the key data set be immediately created on all systems.
Once image copies encrypted by a key are no longer referenced in the local and remote BMCXCOPY tables and will not be used by RECOVER PLUS for recovery as a non-registered image copy, the key is no longer needed by COPY PLUS, RECOVER PLUS, or Log Master and you can eliminate the key.

Key destruction steps are:

1. Delete backups of the current key data set on both the local and remote systems.

2. Remove the row containing the key from the local key data set and duplicate key data sets on remote systems.

   Never remove a row from the key data set unless it is the last row in the data set.

3. Create backups of the new key data set on the local and remote systems.

If a key data set is lost or corrupted and not recoverable, you can gain emergency access to the current key data set with a technique called *key escrow*. Once you have created or updated a key data set, the contents are divided into two or more partial key data sets so that no one data set is sufficient to decrypt an image copy. Each partial key data set is sent to different trusted agent. In the event of an emergency, you can retrieve and reassemble the partial data sets.

Registration of copies

The section describes the registration of encrypted copies.

**Registration for encrypted image copies**

Because the encrypted image copies produced by COPY PLUS are non-standard, encrypted image copies are registered in the BMCXCOPY table (page 582). An STYPE value of e indicates that the image copy is encrypted. In a recovery that uses these encrypted image copies that are registered in BMCXCOPY, you must use RECOVER PLUS.

**Registration for plaintext image copies**

Plaintext full image copies are registered in SYSCOPY. Instant Snapshot copies and certain index space copies are exceptions and are registered according to the rules for BMCXCOPY. (For more information, see the *COPY PLUS for DB2 Reference Manual*.)

A plaintext incremental is registered in SYSCOPY if the most recent primary full copy of the same site type is also plaintext. If the most recent primary full copy of the same site type is encrypted, the incremental is registered in BMCXCOPY.
You can use RECOVER PLUS or the DB2 RECOVER utility to recover using plaintext copies that are registered in SYSCOPY. But you must use RECOVER PLUS for recovery if the plaintext image copies are registered in BMCXCOPY.

Handling DB2 versioning information

When you move objects that contain system pages from one DB2 subsystem to another DB2 subsystem, the version information on the target DB2 subsystem must match the version information on the source DB2 subsystem. If the version information does not match, you cannot access the data on the target DB2 subsystem.

If your version information does not match, use the following steps to move objects to another DB2 subsystem and to ensure that the version information matches:

1. Ensure that the object definitions on the source and target DB2 subsystems are the same. For a table space, each table must have the same number of columns, and each column must be the same data type.

   **TIP**
   
   Use the same ALTER TABLE statement on both the source and target objects.

2. If you are copying indexes that have not been altered in DB2 Version 8, check the SYSIBM.SYSINDEXES catalog table on both subsystems to ensure that the value in both the CURRENT_VERSION column and the OLDEST_VERSION column is 0.

3. If the object has been altered since its creation and has never been reorganized, run a REORG utility on the object. You can determine if an object has been altered but not reorganized by checking the following:

   - If OLDEST_VERSION in SYSIBM.SYSTABLESPACE or SYSIBM.SYSINDEXES is greater than 0, then the space has been reorganized.
   - For table spaces, if OLDEST_VERSION is 0, print the header page of the table space using DSN1PRNT. If the header page has a hash bucket with HPG1V=’00’X as shown below, the space has been altered and has not been reorganized, so you need to run a REORG utility.

```
HASH_BUCKET:  HPGDBKRT='04'X  HPGIBFLG='0D'X
HPGDCOLL='0005'X  HPGBID='0003'X  HPG1V='02'X  HPG1RID='00000000401'X
HPGDCOLL='0006'X  HPGBID='0003'X  HPG1V='01'X  HPG1RID='00000000302'X
HPGDCOLL='0007'X  HPGBID='0003'X  HPG1V='00'X  HPG1RID='00000000301'X
```
Handling DB2 versioning information

For index spaces, if OLDEST_VERSION is 0, print the directory page of the index space using DSN1PRNT. If DIXDCVER is greater than 0 and DIXDOVER = '00'X, the space has not been reorganized and you need to run a REORG utility.

| IX DIRECTORY IXD HEADER:  DIXDHEYE='XD'  DIXDPREFLEN='0050'X  DIXDRootPG='000000
DIXDFLGS='88000000'X DIXDRCID='0003
DIXDDLF='003'X DIXDOLTH='0003'X DIXDKCNM='0001'X
DIXDMLVL='00'X DIXDCVER='00'X DIXDSPAD=' ' DIXDDPAD
DIXDOVER='00'X DIXDB0DV='0000'X DIXDVA0M='01'X DIXD
DIXDLMVER='00'X DIXDSTCK='C0B5256F46FA4301'X |

**TIP**
For DB2 Version 9 and later, if you do not own a REORG utility or you cannot run REORG, ensure that you do an insert after the last ALTER to force the creation of a system page.

4 Ensure that enough version numbers are available. For a table space, the combined active number of versions for the object on both the source and target subsystems must be less than 255. For an index, the combined active number of versions must be less than 16. Use the following guidelines to calculate the active number of versions for the object on both the source and target subsystems:

- If the value in the CURRENT_VERSION column is less than the value in the OLDEST_VERSION column, add the maximum number of versions (255 for a table space or 16 for an index) to the value in the CURRENT_VERSION column.

- Use the following formula to calculate the number of active versions:

\[
\text{number of active versions} = \text{MAX} (\text{target.CURRENT\_VERSION, source.CURRENT\_VERSION}) - \text{MIN} (\text{target.OLDEST\_VERSION, source.OLDEST\_VERSION}) + 1
\]

If the number of active versions is too high, you must reduce the number of active versions by running a REORG utility on both the source and target objects. Then, use the COPY utility to take a copy, and run MODIFY RECOVERY to recycle the version numbers.

5 Run RECOVER PLUS on the target subsystem with the OBIDXLAT option. On the control statement, specify the proper mapping of table database object identifiers (OBIDs) for the table space or index from the source to the target subsystem.

**NOTE**
If you need to rebuild indexes on your target system, do not do so until after you have completed step 6.
6  Run DB2 REPAIR VERSIONS on the object on the target subsystem or specify the RECOVER PLUS UPDATE VERSIONS option in step 5.

This action updates the version numbers in the target system’s catalog using the version numbers in the system pages that have been laid down from the source system and the version numbers on the target system.

For table spaces, the utility updates the following columns:

- OLDEST_VERSION and CURRENT_VERSION in SYSTABLEPART
- VERSION in SYSTABLES
- OLDEST_VERSION and CURRENT_VERSION in SYSTABLESPACE

For indexes, the utility updates OLDEST_VERSION and CURRENT_VERSION in SYSINDEXES. DB2 uses the following formulas to update these columns in both SYSTABLEPART and SYSINDEXES:

\[
\text{CURRENT\_VERSION} = \max(\text{target.CURRENT\_VERSION}, \text{source.CURRENT\_VERSION})
\]
\[
\text{OLDEST\_VERSION} = \min(\text{target.OLDEST\_VERSION}, \text{source.OLDEST\_VERSION})
\]

7  If you need to rebuild indexes on your target system, you can do so now.
Recovering cloned objects

RECOVER PLUS supports the recovery of cloned tables that were introduced in DB2 Version 9. Use the CLONE option on the following RECOVER PLUS commands to recover with clone objects:

- RECOVER (page 189)
- REBUILD (page 221)
- RECOVER UNLOADKEYS (page 232)
- RECOVER BUILDINDEX (page 241)
- LOGSCAN (page 255)

When you are creating an output copy or a model DCB, you can use the symbolic variable &INST for the instance number (page 147, page 180, and page 455). Valid values are 1 and 2.

A clone table is a table with the exact same attributes as an existing table, which is called the base table. The clone is created in the same table space as the base table, is structurally identical to the base table in every way and has the same indexes, triggers, and LOB objects. You can only create clone tables in a universal table space that is managed by DB2.

The base and clone table data and their index data reside in different data sets. For example:

- `catName.DSNDBx.dbName.psName.x0001.A001`
  
  This data set contains instance number 1 (the 1 of x0001).

- `catName.DSNDBx.dbName.psName.x0002.A001`
  
  This data set contains instance number 2 (the 2 of x0001).

DB2 handles log records and log ranges for both base and clone table objects. The base and clone objects are differentiated by differences in the PSID value. The high order bit of the PSID is used to refer to a particular data set instance number as follows:

- a high order bit value of 0 indicates instance number 1
- a high order bit value of 1 indicates instance number 2

Other than the high order bit, the PSID numbers are the same for both a base object and it’s clone.
RECOVER PLUS support for clone objects includes the following limitations:

- You cannot specify the same object with and without the CLONE option in the same SYSIN data set.

RECOVER PLUS issues an error message if you specify a recovery command for the base and clone instance of the same object in the same SYSIN data set.

- All related objects (indexes and underlying table space, LOB objects) should have the same CLONE specification in the same step.

RECOVER PLUS issues an error message if the CLONE specification does not match for the related objects.

- You cannot recover a clone or base object to a point prior to the last EXCHANGE statement.

## Recovering XML objects

RECOVER PLUS supports the recovery of XML objects that were introduced in DB2 Version 9. RECOVER PLUS also supports REBUILD requests for NodeID indexes.

When an XML column is created, DB2 implicitly creates the following XML objects to support the XML column:

- DocID column in the base table - created on the base table and uniquely represents each row.
- DocID index
- XML table
- XML table space containing the XML table
- NodeID index on the XML table

For a point-in-time recovery, you must recover the following XML-related objects to the same point:

- base table — contains the base table, where the logical XML column is stored
- XML table space — contains the XML auxiliary table, where the data is physically stored
- NodeID index on the XML table
- DocID index
Recovering compressed indexes

NOTE

In the CREATE statement and in the DB2 catalog, XML indexes are defined on the XML base table. However, the XML indexes are actually on the XML table (table type ‘P’).

Because of this, the IBM RECOVER utility does not process XML indexes in INDEX(ALL) TABLESPACE baseTableSpace requests. In order to process XML indexes with an INDEX(ALL) request, you must specify TABLESPACE XMLTableSpace.

For XML indexes, RECOVER PLUS behavior is the same as that of IBM.

Recovering compressed indexes

DB2 Version 9 introduced compression for indexes. RECOVER PLUS version 9.1 and later provides native support for recovering copies of compressed indexes made by DB2 or COPY PLUS. (COPY PLUS version 9.1 and later provides native support for copying compressed indexes.) Additionally, RECOVER PLUS provides native support for

- backout recovery of compressed indexes
- recovery of encrypted compressed indexes
- recovery of cabinet copies of compressed indexes
- optimized merge recovery of compressed indexes
- rebuilding compressed indexes

Two methods are available to make copies of compressed indexes. Each method has advantages and disadvantages.

- make copies of compressed indexes by expanding them

Both the DB2 COPY utility and COPY PLUS can make copies of compressed indexes by reading the compressed indexes directly from disk and expanding them before writing the image copy. This method has the advantage that the copy is registered in SYSCOPY (and, in the case of COPY PLUS, the copy is compatible with those produced by the DB2 COPY utility).

Some disadvantages of this method are that it takes time to expand the pages and the resulting image copies are larger than necessary. Also, COPY PLUS copy techniques, such as Instant Snapshot copies and Online Consistent Copies, are not supported. (COPY PLUS will make unexpanded copies if you request an expanded copy.)
make copies of compressed indexes without expanding them

COPY PLUS can make copies of the compressed indexes without expanding them. COPY PLUS registers these copies in the BMCXCOPY table. This method has the advantages that making the copies is faster and the copies are smaller than when copies are made of expanded indexes. Also, all COPY PLUS copy techniques, such as Instant Snapshots, Online Consistent Copies, encrypted copies, and cabinet copies, are supported.

When you make copies of compressed indexes without expanding them, the copy is not compatible with the DB2 utilities and RECOVER PLUS is required to use these copies for recovery.

RECOVER PLUS supports both expanded and unexpanded copies of compressed indexes, and automatically determines the type of the copies that are available.

NOTE
When you use the OUTCOPY option in RECOVER PLUS, copies of compressed indexes must be unexpanded.

When you use COPY PLUS, the IXEXPAND option indicates which method you want to use to create copies of compressed indexes. Valid values are AUTO, YES, and NO. The default value is AUTO. IXEXPAND AUTO specifies that COPY PLUS will make unexpanded copies of compressed indexes if you are running with a Recovery Management solution password (and consequently have RECOVER PLUS to recover using the unexpanded copies). If you are not running Recovery Management, COPY PLUS will make expanded copies for compatibility with IBM. IXEXPAND NO specifies that compressed index copies are unexpanded. IXEXPAND YES specifies that compressed index copies are expanded. For more information on the IXEXPAND option, see the COPY PLUS for DB2 Reference Manual.

NOTE
If you do not have the Recovery Management solution, but you do have the RECOVER PLUS and COPY PLUS products, you should consider changing the value of the COPY PLUS IXEXPAND installation option to NO to achieve the benefits of copying the compressed indexes without expansion.
Using BMC RECOVERY MANAGER groups

You can set up groups in the BMC RECOVERY MANAGER for DB2 product. RECOVER PLUS allows you to identify both table spaces and index spaces in groups for processing by RECOVER PLUS.

Using dynamic grouping, RECOVER PLUS reads the objects in the group each time you run the RECOVER PLUS job, so objects may be added or removed from the group between RECOVER PLUS job runs.

**NOTE**
RECOVER PLUS does not read RECOVERY MANAGER recovery options for the group. If RECOVERY MANAGER recovery options change, you must run RECOVERY MANAGER to pick up the new options and generate the control cards.

Once the group is defined using RECOVERY MANAGER, RECOVER PLUS uses the RECOVERY MANAGER repository and the BMC Common DB2 repository to identify the objects in the group. RECOVER PLUS uses the objects directly from the repository tables. The objects in the group are kept up to date with dynamic grouping.

Dynamic grouping resolves the table space and index object names for inclusion with the RECOVER PLUS commands that support group object types.

RECOVER PLUS provides the OBJECTSET syntax to specify a group. The long range plan is that OBJECTSET will be used as common syntax by other BMC utilities for DB2. You can use OBJECTSET with the following RECOVER PLUS commands:

- RECOVER
- SIMRCVR
- REBUILD INDEX
- SIMBLD INDEX
- LOGSCAN

For more information, see the syntax descriptions for each command in Chapter 3, “RECOVER PLUS syntax”, and specifically, “OBJECTSET specification” on page 176.

Following are some examples of the use of the OBJECTSET syntax:
When you use OBJECTSET, if RECOVER PLUS tries to recover a group of objects and more than 10 of the objects are unrecoverable, RECOVER PLUS issues an error for each object that is unrecoverable and discontinues further processing of the group. This behavior is based on the ERRCONT option that has a default value of 10. To avoid this behavior, set the ON ERROR CONTINUE option with a value that will exceed the possible number of errors in the group, and the recovery will complete.

For more information about grouping in RECOVERY MANAGER, see the RECOVERY MANAGER for DB2 User Guide.

### Syntax

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Objects copied</th>
</tr>
</thead>
</table>
| RECOVER OBJECTSET creator.name<sup>a</sup> | recovers all objects in the object set—table spaces and indexes  
Note that INDEXES YES is not valid with this specification. |
| RECOVER TABLESPACE OBJECTSET creator.name<sup>a</sup> | recovers only table spaces in the object set |
| RECOVER TABLESPACE OBJECTSET creator.name INDEX YES<sup>a</sup> | recovers table spaces in the object set along with their associated indexes, regardless of whether the indexes are included in the group |
| RECOVER INDEX OBJECTSET creator.name<sup>a</sup> | recovers only indexes in the object set |
| REBUILD INDEX OBJECTSET creator.name<sup>b</sup> | rebuilds all indexes in the object set |
| LOGSCAN OBJECTSET creator.name | log scan processing of all objects in the object set—table spaces and indexes |
| LOGSCAN TABLESPACE OBJECTSET creator.name | log scan processing for only table spaces in the object set |
| LOGSCAN TABLESPACE OBJECTSET creator.name INDEX YES | log scan processing for table spaces in the object set along with their associated indexes, regardless of whether the indexes are included in the group |
| LOGSCAN INDEX OBJECTSET creator.name | log scan processing for only indexes in the object set |

<sup>a</sup> SIMRCVR is also valid.

<sup>b</sup> SIMRBLD is also valid.

---

**NOTE**

DSNUM cannot be used with OBJECTSET. The DSNUM used for each object is its DSNUM in the repository.
Recovering a dropped table space or table

This chapter presents the following topics:

Overview ................................................................. 479
Example 1: Recovering a dropped table space when image copies are available 482
Example 2: Recovering a dropped table in a segmented table space containing multiple tables ................................................. 487
Example 3: Recovering a dropped table from a simple table space .................. 491
Example 4: Recovering a dropped table space using a pack backup .................. 494

Overview

This chapter provides an overview and examples for the recovery of a dropped table space or table.

The following considerations apply to drop recovery:

- Drop recovery for indexes is not supported.
- When you are performing a drop recovery, the source definition must match the target definition, including the number of partitions. For partition-by-growth universal spaces, if the number of partitions for the source has grown to more than one partition, the source would no longer match the target, and the drop recovery fails.
If the dropped space was versioned, you must recreate the space with the same versions, or in some cases, you can use DB2 REPAIR VERSIONS.

— If the dropped space has been reorganized after the first alter, you can recreate the space to match the latest version of the space, recover the space, and then run DB2 REPAIR VERSIONS. You cannot rebuild indexes until after you run DB2 REPAIR VERSIONS.

— If the dropped space has not been reorganized, you must recreate the space as it existed originally, before any alters, perform the alters to create the same versions that the dropped space had, and then recover the space.

For more information, see “Handling DB2 versioning information” on page 469.

Figure 112 shows how you can use the RECOVER PLUS product to recover a dropped table space.

To recover a dropped table space, you must first re-create the DB2 table space and index structures. To simplify the task of re-creating objects, you can use the BMC CATALOG MANAGER, ALTER, or CHANGE MANAGER for DB2 products.
RECOVER PLUS rebuilds the data in the table space as it existed at the time of the drop by

- using the OBIDXLAT clause to translate the old OBIDs
- using the INCOPY option for image copies that are now unregistered as input
- using the NOSYSLGRNG option to bypass the SYSLGRNX table when determining which log to scan
Example 1: Recovering a dropped table space when image copies are available

A DBA new to your company has accidentally dropped a production table space instead of the test table space that he intended to drop. You have been assigned to recover this table space to the point in time at which it was dropped. You have already identified the most recent image copy data set.

NOTE
Use the procedure in this example when the table space has been dropped. If you want to recover a table from a table space that has more than one table in it, see “Example 2: Recovering a dropped table in a segmented table space containing multiple tables” on page 487. If you want to recover a table that is the only table in the table space, see “Example 3: Recovering a dropped table from a simple table space” on page 491.

1 Obtain data definitions.

Obtain the definitions of the table space and table that you want to recover. Also, obtain definitions for the indexes because they have also been dropped. You cannot change any characteristics of the table space or table that affect their internal structure.

NOTE
You can obtain the object definitions by using a tool such as one of the following BMC products: CATALOG MANAGER for DB2, ALTER for DB2, or CHANGE MANAGER for DB2. (If you use CHANGE MANAGER for DB2, it can systematically change object names using a name mask in a migrate profile, which could be helpful if many table spaces are involved.)

The data definition language (DDL) for this example is shown in Figure 113 on page 483.
Chapter 7  Recovering a dropped table space or table  483

Example 1: Recovering a dropped table space when image copies are available

2  Create DB2 objects.

Using the DDL from step 1 on page 482, create the table space, table, and indexes. The table space and table must be defined exactly as they were before the drop.

NOTE

If you create the table space more than once (for example, if it is not created correctly the first time), an extra step is required to accomplish the recovery. You must specify a TORBA/TOLOGPOINT value in the RECOVER PLUS control cards to tell RECOVER PLUS when to stop applying log records. Use DSN1LOGP to identify the log point of the first CREATE table space after the drop. This is the log point you should specify with the TORBA/TOLOGPOINT option. If you do not specify a TORBA/TOLOGPOINT value in this case, the recovery will result in an empty table space.

3  Obtain internal object identifiers for the dropped objects.

To run a recovery using the image copy of the dropped table space, you need the internal object identifiers assigned to the database, table space, and table when they were originally created. Because the table space has been dropped, the IDs for the table space and table are no longer available in the DB2 system catalog. You can use the BMC Log Master for DB2 product to find SYSCOPY entries and IDs.

You can also find the object identifiers by using the IBM DB2 stand-alone utility DSN1PRNT to print a range of pages from the image copy data set. The JCL to run DSN1PRNT is shown in Figure 114 on page 484.
Example 1: Recovering a dropped table space when image copies are available

Figure 114 Example JCL to run DSN1PRNT
//RUNPRNT EXEC PGM=DSN1PRNT,
//
PARM='FORMAT,FULLCOPY,PRINT(000,003)'
//STEPLIB
DD DSN=SYS2.DB2.PROD.DSNLOAD,DISP=SHR
//
DD DSN=SYS3.DBAJ.DSNEXIT,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUT1
DD DSN=RDAMSM.ICDBAJ.CRITTS01.G0003V00,
//
DISP=SHR

<== IMAGE COPY DSN

Output from this example job is shown in Figure 115.
Figure 115 Example DSN1PRNT output with the object identifiers of the dropped table space
1DSN1999I START OF DSN1PRNT FOR JOB RDAMSMPR RUNPRNT
DSN1998I INPUT DSNAME = RDAMSM.ICDBAJ.CRITTS01.G0003V00
HEADER PAGE:

PGCOMB='10'X

PGFLAGS='38'X

PGLOGRBA='000000000000'X

HPGOBID='014C0002'X

HPGTORBA='000000000000'X
HPGFOID='0001'X
HPGNUMCO='0004'X
SPACE MAP PAGE:
PGFLAGS='30'X

HPGSGSZ='0008'X

HPGZNUMP='00'X
HPGFLAGS='00'X

HPGSGNAM='SYSDEFLT'

HPGTBLC='0001'X

HPGSSNM='DBAJ'

HPGPARTN='0000'X
HPGROID='0003'X

HPGRBRBA='0003082F5630'X

PGLOGRBA='000000000000'X

SEGNUM='0152'X

PGNUMBER='000000'X

HPGREL='C6'X

HPGMAXL='0039'X

HPGCONTM='19941212081533515711'X

HPGVCATN='DBAJCAT '

PGCOMB='00'X

SEGLENT='00000001'X

PGLOGID='FF'X

HPGHPREF='00000001'X

HPGTSTMP='00010101000000000000'X

HPGPGSZ='1000'X

HPGZPNUM='000000'X

, SEQ

SEGFREE='0151'X

FOEND='N'

PGLOGID='FF'X

SEGENT='0002'X

PGNUMBER='000001'X

SEGSIZE='0008'X

FOEND='E'

FIRST PART OF SEGMENTED SPACE MAP:
SEG 0001

000000000003C0 30000000

SECOND PART OF SEGMENTED SPACE MAP:
RELPG

00

0000

00000000

20

DATA PAGE:

PGCOMB='10'X

PGFREE=3034

60

80

PGLOGRBA='000000000000'X

PGFREE='0BDA'X

PGMAXID='10'X
PGTAIL:

40

PGFREEP=1028

A0

C0

PGLOGID='FF'X

PGFREEP='0404'X

E0

PGNUMBER='000002'X

PGFLAGS='00'X

PGHOLE1='0000'X

PGNANCH=0

PGIDFREE='00'X

PGEND='N'

ID-MAP FOLLOWS:
01

0014 0053 0092 00D1 0110 014F 018E 01CD

09

020C 024B 028A 02C9 0308 0347 0386 03C5

RECORD:

OFFSET='0014'X

PGSFLAGS='00'X

PGSLTH=63

PGSLTH='003F'X

PGSOBD='0003'X

80002711 D1D6C540 40404040 40404040 40404040 40404040 E2D4C9E3 C8404040

PGSID='01'X

....JOE SMITH

40404040 40404040 40404040 40404040 40404040 4040C9D5 C4
RECORD:

OFFSET='0053'X

PGSFLAGS='00'X

PGSLTH=63

PGSLTH='003F'X

IND
PGSOBD='0003'X

80002712 D1C1D5C5 40404040 40404040 40404040 40404040 C4D6C540 40404040
40404040 40404040 40404040 40404040 40404040 4040D4C7 D9

484

RECOVER PLUS for DB2 Reference Manual

PGSID='02'X

....JANE DOE
MGR


Example 1: Recovering a dropped table space when image copies are available

In the output from DSN1PRNT in Figure 115 on page 484, examine the following fields to find the relevant object identifiers:

- **HPGOBID**: The first two bytes of this field are the hexadecimal representation of the database identifier (DBID) of the database (X’014C’ in the example output displayed in Figure 115 on page 484). The last two bytes are the hexadecimal representation of the page set identifier (PSID) of the table space (X’0002’ in the example output displayed in Figure 115 on page 484).

- **PGSOBD**: This field is found in the header information for each row. It is the hexadecimal representation of the object identifier (OBID) of the table for that row (X’0003’ in the example output in Figure 115 on page 484).

**NOTE**

In this example, there is only one table. Because it is a segmented table space, the OBID for the table is also found in the space map segment entries. If your table space has several tables, you may have to print several pages to locate all of the IDs so that you can review the lengths of data in order to distinguish between the tables.

4 Obtain internal object identifiers for the newly created objects.

You can use SPUFI to query the DB2 system catalog for the new internal object identifiers. See Figure 116 for an example.

You can also accept the default values for the target DBID and PSID, as well as for the OBID if you are working with a single-table table space.

**Figure 116  Example SPUFI to locate new internal object identifiers**

```
SELECT DBID,PSID
FROM SYSIBM.SYSTABLESPACE
WHERE DBNAME = 'XYZDB01'
AND NAME   = 'CRITTS01';

---------+---------+---------+---------+---------+---------+
DBID    PSID
---------+---------+---------+---------+---------+---------+
332      12

DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100

SELECT NAME,OBID
FROM SYSIBM.SYSTABLES
WHERE DBNAME = 'XYZDB01'
AND TSNAME = 'CRITTS01';

---------+---------+---------+---------+---------+---------+
NAME                  OBID
---------+---------+---------+---------+---------+---------+
CRITTB01                13

DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```
Example 1: Recovering a dropped table space when image copies are available

5 Create RECOVER PLUS JCL.

Use the DROPRECOVERY and OBIDXLAT options with RECOVER PLUS to accomplish this recovery. Use the OBIDs from the old image copy as source OBIDs and the OBIDs from the newly created object as target OBIDs. Specify the log point of the image copy as the starting point for RECOVER PLUS to apply log records. If you do not know the log point of the image copy, use the value of HPGRBRBA in the output from DSN1PRNT. (For an example, see Figure 115 on page 484.)

NOTE
If the table space was not updated for a long time after the copy was made, the log point value might be too low. A low value is harmless but causes more log to be read. If this is the case, you can compare values in the output of PRINT LOG MAP to obtain an approximate log point. Ensure that the log point you use is not too high.

The RECOVER PLUS JCL and control statements used for this example are shown in Figure 117.

Figure 117 Example drop recovery RECOVER PLUS JCL

```sql
/*===================================================================*/
/*                 RECOVER THE DROPPED TABLESPACE                 */
/*                 USING IMAGE COPY INPUT AND LOG RECORDS        */
/*===================================================================*/
//RECOVER EXEC PGM=AFRMAIN,
       REGION=0M,
       PARM='DBAJ,CRITRECOV,NEW,MSGLEVEL(1)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
//       DD DISP=SHR,DSN=DB2.DSNEXIT
//       DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSUT1 DD UNIT=SYSDA,SPACE=(CYL,(2,1))
//SYSIN DD *
RECOVER TABLESPACE XYZDB01.CRITTS01
DROPRECOVERY
   INCOPY
   FULL DSNAME
   RDAMSM.ICDBAJ.CRITTS01.G0003V00
   LOGPOINT X'0003082F5630'
   OBIDXLAT OBID(X'014C',X'014C')
   PSID(X'0002',X'0000C')
   OBID(X'0003',X'0000D')
REBUILD INDEX(ALL) TABLESPACE XYZDB01.CRITTS01
/*

6 Run the recovery.

The job should end with a condition code 4 and a warning indicating that an image copy is required for the table space. If you would like to create a copy at recovery time, you can add OUTCOPY YES to the recovery and avoid the condition code 4.
Example 2: Recovering a dropped table in a segmented table space containing multiple tables

One of your production table spaces contains multiple tables. Approximately a week ago one table was dropped from the table space. Since then, the other tables in the table space have been updated.

Your challenge is how to recover this dropped table without regressing the updates that occurred to the other tables.

Using the RECOVER PLUS OBIDXLAT and INDEPENDENT OUTSPACE options, your problem is solved. To preserve the updates to the other tables, this process recovers the entire original table space into a temporary table space at the point just prior to dropping the table. The rows from the temporary table space table that corresponds to the dropped table will then be inserted into the newly created table within the original table space.

1 Obtain data definitions.

Obtain the definitions for the table space and all of its tables. If you also want to create the indexes, obtain the definitions of the indexes. (Though it is not necessary to include the indexes in this process, you might want to create them so that you can more easily execute verification queries later.)

NOTE
You can obtain the object definitions by using a tool such as one of the following BMC products: CATALOG MANAGER for DB2, ALTER for DB2, or CHANGE MANAGER for DB2.

2 Create DB2 objects.

Using the DDL you obtained in step 1, create the temporary table space with table and table column definitions identical to the original table space. (Do not change any characteristics of the table space that affect its internal structure.) Also, create the dropped table, along with its indexes, within the original table space.
Obtain the DBID and PSID of the original table space. Obtain the OBIDs of each of the tables within the table space, including the OBID of the table that was dropped. To obtain this information, print the first two pages of the image copy by using the IBM DB2 stand-alone utility DSN1PRNT. For an example of DSN1PRNT JCL, see Figure 118. Figure 119 shows an example of DSN1PRNT output.

The first two bytes of HPGOBI0D are the hexadecimal representation of the DBID (X’014D’ in the example output in Figure 119). The last two bytes of HPGOBI0D are the hexadecimal representation of the PSID of the table space (X’000A’ in the example Figure 119).

---

**Example JCL to run DSN1PRNT to obtain object identifiers for original table space**
```plaintext
//DSN01 EXEC PGM=DSN1PRNT,PARM='PRINT(0,1),FORMAT'
//STEPLIB DD DSN=SYS2.DB2.PROD.DSNLOAD,DISP=SHR
// DD DSN=SYS3.DB2R.DSNEXIT,DISP=SHR
//SYSPRINT DD SYSOUT=* 
//SYSUT1 DD DISP=SHR,DSN=RDADMB.COPY8.G0028V00 <= IMAGE COPY DSN
```

---

**Example DSN1PRNT utility output with object identifiers for original table space**
```
DSN1999I START OF DSN1PRNT FOR JOB RDADMB08 DSN01 
DSN1998I INPUT DSNAME = RDADMB.COPY8.G0028V00 , SEQ 
 HEADER PAGE: PGCOMB='00'X PGLOGRBA='0000000000000000'X PGLOGID='FF'X PNUMBER='000000'X 
 PGFLAGS='38'X HPGOBI0D='014D000A'X HPGOBI0F='00000001'X HPGREL='C6'X 
 HPGTORMB='0000000000000000'X HPGTSTMP='00010101000000000000'X HPGSNN='DBAJ' 
 HPGFOID='0009'X HPGPGBK='1000'X HPGSNG='0010'X HPGPARTN='0000'X 
 HPGZPNUM='00000000'X HPGZNUMP='00000000'X HPGZTBL='0000'X 
 HPGBNOK='0000'X HPGFLG='0000'X HPGCONT='1994121512375917'X 
 HPGGNSN='00000000'X HPGVRAT='DBAJCAT ' HPRBRBA='00003509C1EAE'X FOEND='E' 
 SPACE MAP PAGE: PGCOMB='10'X PGLOGRBA='00003509C159D'X PGLOGID='01'X 
 PNUMBER='000000'X 
 PGFLAGS='30'X SEGNUM='0000000000000000'X SEGFREE='0000000000000000'X SEGENT='0000000000000000'X 
 SEGSIZE='0000000000000000'X FOEND='N' 
 FIRST PART OF SEGMENTED SPACE MAP: 
 SEG 0001 0000000000000000 3000000000000000 
 SEG 0002 0000000000000000 3333333333333333 
 SEG 0003 0000000000000000 3000000000000000 
 SECOND PART OF SEGMENTED SPACE MAP: 
 RELPG 00 20 40 60 80 A0 C0 E0 
 0000 00000000 00000000 
 DSN1994I DSN1PRNT COMPLETED SUCCESSFULLY, 00000002 PAGES PROCESSED 
```
Notice that the OBIDs X’000B’, X’000E’ and X’0011’ are found in the segment entries. By comparing these to the OBIDs in SYSIBM.SYSTABLES for this table space, you can determine which tables are still present and which table has been dropped.

4 Obtain internal object identifiers for the temporary table space.

Obtain the DBID and PSID values of the temporary table space. Obtain the OBID values of each of the tables within this table space. You can query the DB2 system catalog to obtain these identifiers. For an example query, see Figure 120.

**Figure 120** Example query for the DBID, PSID, and OBID values for the temporary table space

```sql
SELECT A.DBNAME, A.TSNAME, A.NAME,
       HEX(A.DBID), HEX(B.PSID), HEX(A.OBID)
FROM SYSIBM.SYSTABLES A,
     SYSIBM.SYSTABLESPACE B
WHERE A.TSNAME = B.NAME
  AND A.DBNAME = B.DBNAME
  AND A.TSNAME = 'DMBSPAC2'
  AND A.DBNAME = 'DMBDROPS'
ORDER BY A.NAME;
```

<table>
<thead>
<tr>
<th>DBNAME</th>
<th>TSNAME</th>
<th>NAME</th>
<th>DBID</th>
<th>PSID</th>
<th>OBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMBDROPS</td>
<td>DMBSPAC2</td>
<td>DMBTBL1</td>
<td>015E</td>
<td>0002</td>
<td>0003</td>
</tr>
<tr>
<td>DMBDROPS</td>
<td>DMBSPAC2</td>
<td>DMBTBL2</td>
<td>015E</td>
<td>0002</td>
<td>0006</td>
</tr>
<tr>
<td>DMBDROPS</td>
<td>DMBSPAC2</td>
<td>DMBTBL3</td>
<td>015E</td>
<td>0002</td>
<td>0009</td>
</tr>
</tbody>
</table>

5 Find the log record sequence number (LRSN) prior to dropping the table.

You can use the DSN1LOGP utility to find the LRSN at which the table was dropped. Because the table was dropped after the last copy, you use the LRSN of this copy for the DSN1LOGP STARTRBA or STARTLRSN. The log output from the DSN1LOGP utility is then searched to find the update to the SYSDBASE table space, DBID(0006) and OBID(0009), which removed the definition for the dropped table. The URID of this event (or for data sharing, the LRSN of the log record at the URID RBA) is used for the TORBA/TOLOGPOINT value in the recovery. Refer to Figure 122 on page 492 and Figure 123 on page 492 for sample JCL and output.

You could also use Log Master to find the LRSN at which the table was dropped.

6 Create and run RECOVER PLUS JCL.

The original table space is recovered into the temporary table space using the RECOVER PLUS INDEPENDENT OUTSPACE option. The OBIDXLAT feature translates the internal identifiers found in the image copy to the identifiers of the temporary table space. Because the table space was not dropped, the SYSCOPY and SYSLGRNX information is still available and the DROPRECOVERY option is not required. For an example of RECOVER PLUS drop recovery JCL, see Figure 121 on page 490.
Example 2: Recovering a dropped table in a segmented table space containing multiple tables

7 Insert rows from the temporary table into the newly created table.

Use SQL to insert the rows from the temporary table into the original table.

```
INSERT INTO ORIGINAL.DMBTBL2
    SELECT * FROM TEMP.DMBTBL2;
```
Example 3: Recovering a dropped table from a simple table space

The table from a simple table space has been accidentally dropped. The table space contains only one table. You are required to recover the table space just prior to when the drop occurred.

1 Obtain data definitions.

Obtain the definition for the dropped table. You cannot change any of the table’s characteristics that affect the internal structure. If you also want to create the indexes, obtain the definitions of the indexes.

NOTE
You can obtain the object definitions by using a tool such as one of the following BMC products: CATALOG MANAGER for DB2, ALTER for DB2, or CHANGE MANAGER for DB2.

2 Create DB2 objects.

Using the DDL obtained in step 1, create the table and indexes.

3 Find the LRSN prior to dropping the table.

You can use the DSN1LOGP utility to find the LRSN at which the table was dropped. Because the table was dropped after the last copy, you supply this LRSN for the DSN1LOGP STARTRBA. You could also use Log Master to find the LRSN.

Search the log output from the DSN1LOGP utility to find the update to the SYSDBASE table space, DBID(0006) and OBID(0009), which removed the definition for the dropped table. Use the URID of this event (or for data sharing, the LRSN of the log record at the URID RBA) for the TORBA / TOLOGPOINT value in the recovery. Locate the time frame of an archive log that would contain the DROP TABLE statement for coding in this example. (DSNJU003 produces a print log map that might help you locate this information.)

Figure 122 on page 492 shows example JCL for the DSN1LOGP utility.
### Figure 122 Example DSN1LOGP utility JCL

```plaintext
//DSNO1 EXEC PGM=DSN1LOGP
//STEPLIB DD DSN=SYS2.DB2.PROD.DSNLOAD,DISP=SHR
// DD DSN=SYS2.DB2.PROD.DSNSAMPLE,DISP=SHR
// DD DSN=SYS3.DBAJ.DSNEXIT,DISP=SHR
//SYSPRINT DD SYSOUT=*  
//SYSSUMRY DD SYSOUT=*  
//ARCHIVE DD DSN=DBAJCAT.ARCHLOG1.A0000567,DISP=SHR
//SYSIN DD *
STARTRBA(00039F08EB02) DBID(0006) OBID(0009)
```

### Figure 123 shows example DSN1LOGP utility output.

---

**SEARCH CRITERIA**

STARTRBA(00039F08EB02)  END RBA (FFFFFFFFFFFFF)
SUMMARY(NO)
DATAONLY(NO)
SYSCOPY(NO)
ALL URIDS ---- YOU MAY SPECIFY URID(XXXXXXXXXXXX)
ALL LUWIDS ---- YOU MAY SPECIFY LUWID(NNNNNNN.LLLLLLL.LLLLLL.LXXXXXXXXXX.XXXX)
DBID(0006)  OBID(0009)
ALL PAGES ---- YOU MAY SPECIFY PAGE(XXXXXXX) MANY TIMES
ALL TYPES ---- YOU MAY SPECIFY TYPE(XX)
ALL SUBTYPES ---- YOU MAY SPECIFY SUBTYPE(XX)

```
DSN1212I DSN1LGRD FIRST LOG RBA ENCOUNTERED 00039F08EB02
```

**(LINES OMITTED)**

<table>
<thead>
<tr>
<th>URID</th>
<th>LRSN</th>
<th>DBID</th>
<th>OBID</th>
<th>PAGE</th>
<th>TYPE</th>
<th>CLR</th>
</tr>
</thead>
<tbody>
<tr>
<td>00039F0A080</td>
<td>AE68AC13F86E</td>
<td>0006</td>
<td>0009</td>
<td>00243</td>
<td>UNDO REDO</td>
<td>NO</td>
</tr>
<tr>
<td>00039F0A959</td>
<td>AE68AC13F871</td>
<td>0006</td>
<td>0009</td>
<td>00243</td>
<td>UNDO REDO</td>
<td>NO</td>
</tr>
</tbody>
</table>

---
Example 3: Recovering a dropped table from a simple table space

4. Create and run RECOVER PLUS JCL.

Recover the table space using the OBIDXLAT option to translate the OBID found in the image copy to the OBID of the newly created table. Because the table space was not dropped, the SYSCOPY and SYSLGRNX information is still available and the DROPRECOVERY option is not required.

RECOVER PLUS generates an image copy to allow for recoverability after the translation. The RECOVER PLUS OUTPUT command causes dynamic allocation of the image copy. (See “OUTPUT” on page 145 for details regarding the OUTPUT command and dynamic allocation of output data sets.) Figure 124 provides example RECOVER PLUS JCL for this step.

Figure 124 Example RECOVER PLUS JCL with dynamically allocated output copy

```plaintext
//RECOVER EXEC PGM=AFRMAIN,REGION=5M,
//      PARM='DBAJ,DMBBASIC,NEW,MSGLEVEL(1)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
//      DD DISP=SHR,DSN=DB2.DSNEXIT
//      DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN    DD *
OUTPUT CP00001
   DSNAME RDADMB.COPY8(+1)
   UNIT SYSDA
   CATLG YES
   MODELDCB SYS1.MODEL
   SPACE (20,1) CYL
RECOVER TABLESPACE DMBDP65.DMBSAC1
   OUTCOPY YES
   OUTCOPYDDN CP00001
   OBIDXLAT OBID( X'3', X'4')
   TORBA X'000343F5F6EC'
```
Example 4: Recovering a dropped table space using a pack backup

A table space at your company has been accidentally dropped. The most recent image copy of this table space is more than three weeks old. However, you know that your DASD management group makes pack backups of all DB2 data sets every night while DB2 is down.

You have been assigned to recover this table space to the point in time at which it was dropped. To avoid processing three weeks of log data, you decide to use the latest pack dump as input.

1 Obtain data definitions.

Using a tool such as the BMC CATALOG MANAGER for DB2, ALTER for DB2, or CHANGE MANAGER for DB2 product, obtain the data definitions of the table space and table you want to recover. Also, obtain definitions for the indexes because they were also dropped. (If you use CHANGE MANAGER for DB2, it can systematically change object names using a name mask in a migrate profile, which could be helpful if many table spaces are involved.) You cannot change any of the characteristics of the table space or table that affect their internal structure. The DDL for this example is shown in Figure 125.

Figure 125  Example DDL for the dropped table space

```
CREATE TABLESPACE CRITTS01 IN ABCDB01
  FREEPAGE 0
  USING STOGROUP SYSDEFLT
  PRIQTY 40 SECQTY 40
  SEGSIZE 8;
COMMIT;
CREATE TABLE ABCDB01.CRITTB01(
  EMPNO     INTEGER  NOT NULL
, FNAME     CHAR(20) NOT NULL WITH DEFAULT
, LNAME     CHAR(30) NOT NULL WITH DEFAULT
, JOBCODE   CHAR(03) NOT NULL WITH DEFAULT 3
)
  IN ABCDB01.CRITTS01;
COMMIT;
CREATE UNIQUE INDEX ABCDB01.CRITIX01
  ON ABCDB01.CRITTB01 (EMPNO)
  USING STOGROUP SYSDEFLT PRIQTY 40 SECQTY 40
  CLUSTER;
COMMIT;
```
2 Create DB2 objects.

Using the DDL from step 1 on page 494, create the table space, table, and indexes. The table space and table must be defined exactly as they were before they were dropped.

NOTE

If you create the table space more than once (for example, if it is not created correctly the first time), an extra step will be required to accomplish the recovery. You must specify a TORBA/TOLOGPOINT value in the RECOVER PLUS job to tell RECOVER PLUS when to stop applying log records. Use DSN1LOGP to identify the log point of the first CREATE TABLESPACE statement after the drop. This is the log point that you should specify with the TORBA/TOLOGPOINT option. If you do not specify a TORBA/TOLOGPOINT value in this case, the recovery will result in an empty table space.

3 Restore the backup.

Using whatever backup and restore mechanism is in place, restore the table space and index data sets.

The example in Figure 126 uses the IBM utility DFDSS to restore the data sets.

Figure 126  Example DFDSS JCL to restore the table space data set

```plaintext
//ADRDSSU EXEC  PGM=ADRDSSU,REGION=0M
//SYSPRINT  DD SYSOUT=*  
//TAPE      DD DSN=RDAMSM.ABCDB01.DUMP, 
            DISP=SHR  
//SYSIN     DD * 
REST -  
DS (   
      INCLUDE(DBAJCAT.DSNDBC.ABCDB01.**) ) - 
      IDD(TAPE) - 
      REPLACE - 
      ODY(DEV165) - 
      CATALOG
/*
```

4 Create RECOVER PLUS JCL.

You need to use the DROPRECOVERY and OBIDXLAT options to accomplish this recovery.

Using the LOGONLY option indicates that there is no image copy input and log records should be applied beginning after the HPGRBRRBA value in the header page of the table space data set. Figure 127 on page 496 shows the example RECOVER PLUS JCL.
5 Run the recovery.

The job should end with a condition code 4 and a warning message indicating that an image copy is required for this table space. If you want to create a copy during the recovery and avoid the condition code 4, you can add OUTCOPY YES to the recovery.

---

**Figure 127  Example RECOVER PLUS JCL for recovery using a pack backup**

```
//RECOVER EXEC PGM=AFRMAIN,REGION=5M,
//            PARM='DBAJ,DMBBASIC,NEW,MSGLEVEL(1)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN DD *
RECOVER TABLESPACE ABCDB01.CRITTS01
  DROPRECOVERY
  OBIDXLAT DBID (X'014F',X'014F')
  PSID (X'002',X'0011')
  OBID (X'003',X'0012')
  LOGONLY
RECOVER INDEXSPACE ABCDB01.CRITIX01
  LOGONLY
```
Migrating data

This chapter presents the following topics:

Overview .......................................................... 497
Defining target objects for migration .......................... 500
Example 1: Creating a consistent database for query and review .............. 501
Example 2: Moving an application to another subsystem with index recovery..... 508
Example 3: Moving an application to another subsystem with index rebuilding . 518

Overview

This chapter provides examples for data migration. The RECOVER PLUS product supports data migration from one DB2 table space to another. Table spaces may reside on the same DB2 subsystem or on different DB2 subsystems or data sharing groups.

NOTE

The Online Consistent Copy component of the Recovery Management for DB2 solution can create a consistent copy without a quiesce by using Instant Snapshots. See the Recovery Management for DB2 User Guide section “Examples of Online Consistent Copy Jobs,” for the following examples:

- Example 2, “Migrating Data by Creating a Consistent Copy and Renaming the Data Sets”
- Example 3, “Recovering a Table Space and Index by Using Online Consistent Copy with RECOVER PLUS”

Also note that there are special considerations for migrating versioned tables. For more information, see “Handling DB2 versioning information” on page 469.
Figure 128 shows how you can use RECOVER PLUS to migrate data from a production table space to a table space on a different subsystem designated for high-level query functions.

**Figure 128  RECOVER PLUS migration within the same complex**

In this example, it is assumed that both DB2 subsystems run on systems that share a common DASD pool.

You use RECOVER PLUS to build a table space and index set for the receiving DB2 system by using copies, change accumulation files, and logs made for recovery purposes on the sending DB2 system. To perform the migration, you use the INDEPENDENT OUTSPACE option and the OBIDXLAT option.

You can migrate the indexes using copies and logs in the same manner as the table space. Alternatively, you can rebuild the indexes by specifying the newly migrated table space with the INDEPENDENT INTABLESPACE option. In either case, the structure of the indexes cannot be different in this scenario. (Some indexes may be eliminated, however.)

Figure 129 on page 499 demonstrates how you can use RECOVER PLUS to migrate data to remote sites. In this example, the recovery resources on the sending subsystem are used to build a reference migration image. A reference migration image is identical to a SHRLEVEL REFERENCE image copy.
One copy of this image copy may be registered at the sending site for use in normal DB2 recovery operations. Another copy could be sent off site to build a query version of the production table space at a remote location.

At the receiving site, you can use the INCOPY option of RECOVER PLUS to restore the reference migration image to the query version of the object.

Indexes can also be migrated using the reference migration image. However, if you rebuild the indexes instead, the index structure may be different on the query system to optimize query access (for example, using QMF).
Defining target objects for migration

In order to successfully migrate data from a source image copy to a target space, the source and target object definitions must be identical. You cannot change any of the characteristics of the table space or tables that affect their internal structure. Target index definitions can differ if the target indexes are to be rebuilt rather than recovered from a source image copy.

**NOTE**

When you are performing a migration, the source definition must match the target definition, including the number of partitions. For partition-by-growth universal spaces, if the number of partitions for the source has grown to more than one partition, the source would no longer match the target, and the migration fails.

You can obtain the object definitions by using a tool such as one of the following BMC products: CATALOG MANAGER for DB2, ALTER for DB2, or CHANGE MANAGER for DB2. (If you use CHANGE MANAGER for DB2, it can systematically change object names using a name mask in a migrate profile, which could be helpful if many table spaces are involved).

It is important to be aware of any alterations to source definitions. For example, assume Source_Table_A has been defined with 5 fixed-length columns. Subsequently, Source_Table_A is altered to add a variable-length column and the source image copy is created before a REORG is executed. Defining Target_Table_A with 6 columns (5 fixed-length and 1 variable-length) and using the source image copy for migration will result in a data incompatibility error (00C90101) when trying to access the target data. In this case, Target_Table_A needs to be defined with 5 fixed-length columns, and then altered to add the variable-length column. (If you use CHANGE MANAGER for DB2, it will identify these types of changes to source objects and build target definition and actions appropriately).
Example 1: Creating a consistent database for query and review

Every evening at midnight, a version of an order and shipping database must be migrated to a query database. To create management reports, you must be able to execute long-running queries on the query database without interfering with online transactions.

Orders and shipping data are added by the online transactions. These transactions cannot interfere with the consistency of a series of queries against the query database that remains static while you create the management reports.

In this example, the databases are defined on the same DB2 subsystem. Your challenge is to create a consistent database for query and review (see Figure 130).

Figure 130 Creating a consistent database for query and review

To create a database that is consistent for query and review, you perform the following steps:

1. Obtain data definitions.

   Obtain the definitions of the database, table spaces, and tables from the online application. These are the source objects. Also, obtain definitions for the indexes if you want to create them.

   ![Diagram](image-url)
### Example 1: Creating a consistent database for query and review

The DDL for this example is shown in Figure 131.

**Figure 131  Data definitions for source objects**

<table>
<thead>
<tr>
<th>DDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE DATABASE AFRORDER;</td>
</tr>
<tr>
<td>CREATE TABLESPACE AFRORDER IN AFRORDER USING STOGROUP PROD1;</td>
</tr>
<tr>
<td>CREATE TABLESPACE AFRSHIPS IN AFRORDER USING STOGROUP PROD1;</td>
</tr>
<tr>
<td>CREATE TABLE AFRORDER (</td>
</tr>
<tr>
<td>ORDERNUM INT</td>
</tr>
<tr>
<td>ITEMCODE INT</td>
</tr>
<tr>
<td>PRICE    DECIMAL (5,2)</td>
</tr>
<tr>
<td>QUANTITY  SMALLINT</td>
</tr>
<tr>
<td>TOTAL    DECIMAL (7,2)</td>
</tr>
<tr>
<td>)</td>
</tr>
<tr>
<td>IN AFRORDER.AFRORDER;</td>
</tr>
<tr>
<td>CREATE TABLE AFRSHIPS (</td>
</tr>
<tr>
<td>ORDERNUM  INT</td>
</tr>
<tr>
<td>DATEORDERED DATE</td>
</tr>
<tr>
<td>DATESHIPPED DATE</td>
</tr>
<tr>
<td>)</td>
</tr>
<tr>
<td>IN AFRORDER.AFRSHIPS;</td>
</tr>
<tr>
<td>CREATE UNIQUE INDEX AFRORDI ON AFRORDER (ORDERNUM)</td>
</tr>
<tr>
<td>USING STOGROUP PROD1;</td>
</tr>
<tr>
<td>CREATE UNIQUE INDEX AFRSHPI ON AFRSHIPS (ORDERNUM)</td>
</tr>
<tr>
<td>USING STOGROUP PROD1;</td>
</tr>
</tbody>
</table>

2 Create the database.

Modify the DDL for the source objects to change the names so that table spaces and tables are distinctly defined. You might want to create a new database as shown in Figure 132 on page 503. Also, consider other possible changes such as the STOGROUP and index type. Figure 132 on page 503 shows the DDL for the target objects.

**NOTE**

See “Defining target objects for migration” on page 500 for more information.
3 Obtain internal IDs.

You need the internal DB2 OBIDs for the source and target objects to set up the migration. You can use SQL to obtain these identifiers from the DB2 system catalog. See Figure 133 for an example.

**NOTE**

You can accept the default values for the source DBID and PSID. If the table space contains only one table, you can also accept the default values for the source OBIDs. RECOVER PLUS extracts this information from the catalog. These IDs are in decimal. Because a new database is created, all of the IDs except for the DBID are the same in this example. They are included in the migration job for documentation purposes.
4 Quiesce the source objects.

Issue a QUIESCE command on the source table spaces. Example JCL and output are shown Figure 134 and Figure 135 on page 505.

Figure 134  Example QUIESCE JCL

```
//QUIESCE EXEC DSNUPROC.SYSTEM=DBAN.UID='AFRRESM1',
// UTPROC='',REGION=4M,
// LIB='SYS2.DB2.PROD.DSNLOAD'
//DSNUPROC.SYSPRINT DD SYSOUT=* 
//DSNUPROC.SYSIN DD *
QUIESCE TABLESPACE AFRORDER.AFRORDER
   TABLESPACE AFRORDER.AFRSHIPS
```
Example 1: Creating a consistent database for query and review

5 Stop the target objects.

Issue a STOP command against the database or table space where the query image will be built.

For example:

```
*STOP DATABASE (AFRSUMOR)
```

6 Run RECOVER PLUS to create the target image.

In this step, you migrate the image to the other spaces using the RECOVER PLUS INDEPENDENT OUTSPACE and OBIDXLAT options.

The data sets that are to receive the migrated data were created when the data definition was done for the target table spaces and index spaces. These data sets are specified in the syntax with their VSAM cluster names.

In the example shown in Figure 136 on page 506, the IDs from Step 3 are used to define the objects required. Note that TORBA (or TOLOGPOINT) LASTQUIESCE is used. If other quiesce points might be created by other activities, use TORBA (or TOLOGPOINT) X’0003D5B9A3C2’—the LRSN from the QUIESCE operation.

---

### Figure 135 Example QUIESCE output

| DSNU000I | DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = LSBRESM1 |
| DSNU050I | DSNUGUTC - QUIESCE TABLESPACE AFRORDER.AFRORDER.AFRORDER.AFRSHIPS |
| DSNU477I | *DSNP DSNUQUIA - QUIESCE SUCCESSFUL FOR TABLESPACE AFRORDER.AFRORDER.AFRSHIPS |
| DSNU477I | *DSNP DSNUQUIA - QUIESCE SUCCESSFUL FOR TABLESPACE AFRORDER.AFRORDER.AFRSHIPS |
| DSNU474I | *DSNP DSNUQUIA - QUIESCE AT RBA 0003D5B9A3C2 AND AT LRSN 0003D5B9A3C2 |
| DSNU475I | DSNUQUIB - QUIESCE UTILITY COMPLETE, ELAPSED TIME= 00:00:00 |
| DSNU010I | DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0 |

---

**NOTE**

- This quiesce will be the end point (TORBA/TOLOGPOINT) of the RECOVER PLUS request. Note the LRSN.

- At this point, the source objects can begin to be updated by other transactions.

- If image copies (even SHRLEVEL CHANGE) can be made just before this quiesce point, less information from the log will be needed to accomplish the migration. As a result, the migration will be completed more quickly.
**Example 1: Creating a consistent database for query and review**

**NOTE**

This step does not affect the source table spaces or indexes.

You can use the RECOVER INDEX command to recover the indexes from image copies and logs if copies of the indexes have been made with COPY PLUS for DB2, DSN1COPY, or the IBM COPY utility.

**Figure 136 Example RECOVER PLUS JCL to create the target image**

```
//RECINDEP EXEC PGM=AFRMAIN,REGION=0M,
// PARM='DBAN,AFRAQDN,NEW,MSGLEVEL(1)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// DD DISP=SHR,DSN=DB2.DSNLOAD
//SORTOUT DD DUMMY
//SYSIN DD *
OPTIONS INDEXLOG YES
RECOVER TABLESPACE AFRORDER.AFRORDER
    OBIDXLAT DBID(352,449) PSID (2,2) OBID (3,3)
    INDEP OUTSPACE MODEL
        DBANCAT.DSNDBC.AFRSUMOR.AFRSUMOR.I0001.A001
    TORBA LASTQUIESCE
RECOVER INDEX OWNERID.AFRORDI
    OBIDXLAT DBID(352,449) PSID (5,5) OBID (3,3)
    INDEP OUTSPACE MODEL
        DBANCAT.DSNDBC.AFRSUMOR.AFRORDSI.I0001.A001
    TORBA LASTQUIESCE
RECOVER INDEX OWNERID.AFRSHPI
    OBIDXLAT DBID(352,449) PSID (10,10) OBID (8,8)
    INDEP OUTSPACE MODEL
        DBANCAT.DSNDBC.AFRSUMOR.AFRSHPSI.I0001.A001
    TORBA LASTQUIESCE
RECOVER TABLESPACE AFRORDER.AFRSHIPS
    OBIDXLAT DBID(352,449) PSID (7,7) OBID (8,8)
    INDEP OUTSPACE MODEL
        DBANCAT.DSNDBC.AFRSUMOR.AFRSUMSH.I0001.A001
    TORBA LASTQUIESCE
```

7 Start the target objects.

Issue a START command to start the target objects. They are now ready for queries.

For example, use the following command:

```
-START DATABASE (AFRUMOR)
```
8 Allow queries.

Queries such as the one shown in Figure 137 (which requires a table space scan) can be completed without interfering with the online processing. Because data will not be changing on this image, the result of the query will be consistent with other queries made until the data is refreshed again.

**Figure 137 Query on target system**

```
SELECT COUNT(*), SUM(TOTAL)
FROM   AFR.AFRSUMORDER ORD,
       AFR.AFRSUMSHIPS SHIP
WHERE  
    (DAYS(DATESHIPPED) > DAYS(DATEORDERED) + 2 OR
     DAYS(DATEORDERED) + 2 < DAYS(CURRENT DATE) AND
     DATESHIPPED IS NULL
    )
AND    ORD.ORDERNUM = SHIP.ORDERNUM;
```

+--------------------------------------+
|               |                    |
+--------------------------------------+
| 1 | 2 | 888.50 |
+--------------------------------------+
Example 2: Moving an application to another subsystem with index recovery

For use in auditing, you need to build a shadow of an application that runs on a non-data sharing system. The audit application begins at the end of the year.

NOTE

In this example, RECOVER INDEX is used; therefore, the source and target indexes must be the same.

The table spaces and index spaces for the shadow application are on another DB2 subsystem that is on another computer system with no shared DASD.

1 Obtain an RBA for a consistent view of the source table spaces.

The table spaces for the application are AFRACCT, AFRTRAN, AFRSUMM, and AFRHIST.

These table spaces are in database AFRFIN. They need to be at a consistent point for the migration. The point at which logging starts after DB2 goes down normally, flushes all buffers, handles all transactions, and comes up again, can be used for a point of consistency. A sample of messages from DB2 logging is shown in Figure 138.

Figure 138  Sample messages from DB2 logging

```
DSNR004I > RESTART...UR STATUS COUNTS
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
DSNR005I > RESTART...COUNTS AFTER FORWARD RECOVERY
IN COMMIT=0, INDOUBT=0
DSNR006I > RESTART...COUNTS AFTER BACKWARD RECOVERY
.
.
.
DSNJ099I > LOG RECORDING TO COMMENCE WITH
_STARTRBA=0004D176C05C
```

NOTE

Rather than having to determine the RBA for the last shutdown, in this example the option TORBA LASTSHUTDOWN is used to allow RECOVER PLUS to select the last shutdown RBA. This technique is used because the system is taken down for year-end synchronization. Another technique for establishing a point of consistency is to issue a QUIESCE command for all of the table spaces in one statement. (For an example, see Figure 134 on page 504.) You could also use the point of consistency from an ARCHIVE LOG MODE (QUIESCE) command.
2 Create migration images.

Run RECOVER PLUS to create migration images. The syntax is completely described in Chapter 3, “RECOVER PLUS syntax.”

In the example JCL shown in Figure 139, you create migration images of the table spaces at the point of the pause of the DB2 subsystem that you captured in step 1 on page 508. These data sets are identical in format to an image copy or DSN1COPY of the spaces. However, you do not register these copies because they are going to be shipped to the audit system. Also, because they are going to a different computer system, you do not catalog these copies with the operating system.

If you want to make registered copies, see Chapter 9, “Making copies from the log and earlier copies.” You may combine these operations.

Note that AFRAACCT is partitioned, but the OUTCOPY ASCODED option is used to make a copy that is for all of the partitions.

Figure 139 Example RECOVER PLUS JCL to create migration images with index recovery (part 1 of 2)

```plaintext
//RECOUTC1 EXEC PGM=AFRMAIN,REGION=0M,
// PARM='DBAN,AFRAQDN,NEW,MSGLEVEL(1)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPICK DD SYSOUT=*
//ACCTCPY DD DSN=AFR.ACCTCPY1,
// UNIT=CART,
// DISP=(NEW,KEEP)
//TRANCPY DD DSN=AFR.TRANCPY1,
// UNIT=CART,
// DISP=(NEW,KEEP)
//SUMMCPY DD DSN=AFR.SUMMCPY1,
// UNIT=CART,
// DISP=(NEW,KEEP)
//HISTCPY DD DSN=AFR.HISTCPY1,
// UNIT=CART,
// DISP=(NEW,KEEP)
//ACCICPY DD DSN=AFR.ACCICPY1,
// UNIT=CART,
// DISP=(,KEEP)
//TRNICPY DD DSN=AFR.TRNICPY1.
// UNIT=CART,
// DISP=(,KEEP)
//SUMICPY DD DSN=AFR.SUMICPY1,
// UNIT=CART,
// DISP=(,KEEP)
//HISICPY DD DSN=AFR.HISICPY1.
// UNIT=CART,
// DISP=(,KEEP)
```
Obtain data definitions.

Obtain the definitions of the database, table spaces, and tables for which you want to create the audit application for querying. These are the source objects. Also, obtain definitions for the indexes if you want to create them in the audit application.

**NOTE**

See “Defining target objects for migration” on page 500 for more information.

You cannot change any of the characteristics of the table space, tables, or indexes that affect their internal structure. The AFRTRAN table space must remain partitioned and use the same key. If you want to change or add to the indexes used, you must use the technique shown in “Example 3: Moving an application to another subsystem with index rebuilding” on page 518.

For this example, you would obtain the DDL shown in Figure 140.
| CREATE TABLE AFRACCT (  
|     ACCTNO INT  
|     ACCTDESC CHAR(30)  
|     ACCTTYPE CHAR(10)  
|     BALANCE DECIMAL (10,2)  
| )  
| IN AFRFIN.AFRACCT;  
| CREATE TABLE AFRTRAN (  
|     ACCTNO INT  
|     TRANSDATE DATE  
|     TRANSDESC CHAR(40)  
|     CLERKID CHAR(4)  
|     TOTAL DECIMAL (10,2)  
| )  
| IN AFRFIN.AFRTRAN;  
| CREATE TABLE AFRSUMM (  
|     ACCTNO INT  
|     REPCAT CHAR (40)  
|     SUMMCAT CHAR(40)  
|     SUMMDATE DATE  
|     TOTAL DECIMAL (10,2)  
| )  
| IN AFRFIN.AFRSUMM;  
| CREATE TABLE AFRHIST (  
|     ACCTNO INT  
|     TRANSDATE DATE  
|     TRANSDESC CHAR(40)  
|     CLERKID CHAR(4)  
|     TOTAL DECIMAL (10,2)  
| )  
| IN AFRFIN.AFRHIST;  
| CREATE INDEX AFRACCTI ON AFRACCT (ACCTNO)  
| USING STOGROUP AFRTEST  
| CLUSTER (  
|     PART 1 VALUES(30000)  
| , PART 2 VALUES(60000)  
| , PART 3 VALUES(99999)  
| );  
| CREATE UNIQUE INDEX AFRTRANI ON AFRTRAN (ACCTNO)  
| USING STOGROUP AFRTEST;  
| CREATE UNIQUE INDEX AFRSUMMI ON AFRSUMM  
| (ACCTNO,REPCAT,SUMMCAT,SUMMDATE)  
| USING STOGROUP AFRTEST;  
| CREATE INDEX AFRHISTI ON AFRHIST (ACCTNO)  
| USING STOGROUP AFRTEST;  
|
4 Obtain object IDs for the source objects.

On the source system, obtain the object IDs. You can do this by using SQL statements to query the catalog or by using some other tool. The example SQL statements and the output are shown in Figure 141.

**NOTE**

You can use the default values for the source DBID and PSID. If the table space contains only one table, you can also use the default values for the source OBIDs. RECOVER PLUS extracts this information from the image copy.

---

**Figure 141 Example SQL and output to obtain object IDs for source objects**

```sql
SELECT NAME, DBID, PSID FROM SYSIBM.SYSTABLESPACE WHERE
    NAME = 'AFRTRAN' OR NAME = 'AFRACCT' OR
    NAME = 'AFRHIST' OR NAME = 'AFRSUMM';
```

<table>
<thead>
<tr>
<th></th>
<th>NAME</th>
<th>DBID</th>
<th>PSID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AFRACCT</td>
<td>321</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>AFRHIST</td>
<td>321</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>AFRSUMM</td>
<td>321</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>AFRTRAN</td>
<td>321</td>
<td>4</td>
</tr>
</tbody>
</table>

```sql
SELECT CREATOR, NAME, INDEXTYPE, DBID, ISOBID, OBID FROM SYSIBM.SYSINDEXES WHERE
    NAME = 'AFRACCTI' OR NAME = 'AFRTRANI' OR
    NAME = 'AFRSUMMI' OR NAME = 'AFRHISTI';
```

<table>
<thead>
<tr>
<th></th>
<th>CREATOR</th>
<th>NAME</th>
<th>INDEXTYPE</th>
<th>DBID</th>
<th>ISOBID</th>
<th>OBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>OWNERID</td>
<td>AFRACCTI</td>
<td>2</td>
<td>321</td>
<td>14</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>OWNERID</td>
<td>AFRHISTI</td>
<td>2</td>
<td>321</td>
<td>20</td>
<td>19</td>
</tr>
<tr>
<td>3</td>
<td>OWNERID</td>
<td>AFRSUMMI</td>
<td>2</td>
<td>321</td>
<td>18</td>
<td>17</td>
</tr>
<tr>
<td>4</td>
<td>OWNERID</td>
<td>AFRTRANI</td>
<td>2</td>
<td>321</td>
<td>16</td>
<td>15</td>
</tr>
</tbody>
</table>

```sql
SELECT NAME, OBID FROM SYSIBM.SYSTABLES WHERE DBNAME = 'AFRFIN';
```

<table>
<thead>
<tr>
<th></th>
<th>NAME</th>
<th>OBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AFRACCT</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>AFRTRAN</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>AFRSUMM</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>AFRHIST</td>
<td>12</td>
</tr>
</tbody>
</table>
```
Create objects on the target system.

Create the database, table spaces, tables, and indexes on the target system using the data definitions in Figure 142. They need to be consistent in structure with the source system. (RECOVER PLUS provides migration checking when INCOPY is used. See the note on page 515 for more information.)

**Figure 142  Data definitions for target objects (part 1 of 2)**

```sql
CREATE DATABASE AFRAUDIT;
CREATE TABLESPACE AFRACCT IN AFRAUDIT USING STOGROUP AFRTEST
    NUMPARTS 3;
CREATE TABLESPACE AFRTRAN IN AFRAUDIT USING STOGROUP AFRTEST;
CREATE TABLESPACE AFRSUMM IN AFRAUDIT USING STOGROUP AFRTEST;
CREATE TABLESPACE AFRHIST IN AFRAUDIT USING STOGROUP AFRTEST;
CREATE TABLE AFRACCT
    (ACCTNO INT,
     ACCTDESC CHAR(30),
     ACCTTYPE CHAR(10),
     BALANCE DECIMAL (10,2)
    ) IN AFRAUDIT.AFRACCT;
CREATE TABLE AFRTRAN
    (ACCTNO INT,
     TRANSDATE DATE,
     TRANSDESC CHAR(40),
     CLERKID CHAR(4)
     TOTAL DECIMAL (10,2)
    ) IN AFRAUDIT.AFRTRAN;
CREATE TABLE AFRSUMM
    (ACCTNO INT,
     REPCAT CHAR (40),
     SUMMCAT CHAR(40),
     SUMMDATE DATE,
     TOTAL DECIMAL (10,2)
    ) IN AFRAUDIT.AFRSUMM;
CREATE TABLE AFRHIST
    (ACCTNO INT,
     TRANSDATE DATE,
     TRANSDESC CHAR(40),
     CLERKID CHAR(4)
     TOTAL DECIMAL (10,2)
    ) IN AFRAUDIT.AFRHIST;
```
Example 2: Moving an application to another subsystem with index recovery

6 Obtain IDs on the target system.

On the target system, obtain the object IDs needed for translation. You can do this by using SQL statements to query the DB2 system catalog. The example SQL statements and the output are shown in Figure 143.

**NOTE**
You can use the default values for the target DBID and PSID. If the table space contains only one table, you can also use the default values for the target OBIDs. RECOVER PLUS extracts this information from the catalog.

---

**Figure 143  Example SQL and output to obtain object IDs for target objects (part 1 of 2)**

```
SELECT NAME, DBID, PSID FROM SYSIBM.SYSTABLESPACE WHERE
    NAME = 'AFRTRAN' OR NAME = 'AFRACCT' OR
    NAME = 'AFRHIST' OR NAME = 'AFRSUMM';

+------------+----------+----------+
| NAME       | DBID     | PSID     |
+------------+----------+----------+
| AFRACCT    | 336      | 2        |
| AFRHIST    | 336      | 8        |
| AFRSUMM    | 336      | 6        |
| AFRTRAN    | 336      | 4        |
+------------+----------+----------+

SELECT CREATOR, NAME, INDEXTYPE, DBID, ISOBID, OBID FROM SYSIBM.SYSINDEXES WHERE
    NAME = 'AFRACCTI' OR NAME = 'AFRTRANI' OR
    NAME = 'AFRSUMMI' OR NAME = 'AFRHISTI';
```
Example 2: Moving an application to another subsystem with index recovery

Chapter 8  Migrating data  515

Recover with INCOPY and ID translation.

The tapes from the source system are delivered at the target system. You use the external label information to complete the JCL to create data for the target. The ID translation is built from the data in step 4 on page 512 and step 6 on page 514. Output copies are made to allow a normal recovery on the audit system, if required.

7 Recover with INCOPY and ID translation.

<table>
<thead>
<tr>
<th>CREATOR</th>
<th>NAME</th>
<th>INDEXTYPE</th>
<th>DBID</th>
<th>ISOBID</th>
<th>OBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>OWNERID</td>
<td>AFRACCTI</td>
<td>2</td>
<td>336</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>OWNERID</td>
<td>AFRHISTI</td>
<td>2</td>
<td>336</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>OWNERID</td>
<td>AFRSUMMI</td>
<td>2</td>
<td>336</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>OWNERID</td>
<td>AFTRANI</td>
<td>2</td>
<td>336</td>
<td>16</td>
</tr>
</tbody>
</table>

SELECT NAME, OBID FROM SYSIBM.SYSTABLES WHERE DBNAME = 'AFRAUDIT';

<table>
<thead>
<tr>
<th>NAME</th>
<th>OBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFRACCT</td>
<td>9</td>
</tr>
<tr>
<td>AFTRAN</td>
<td>10</td>
</tr>
<tr>
<td>AFRSUMM</td>
<td>11</td>
</tr>
<tr>
<td>AFTRHIST</td>
<td>12</td>
</tr>
</tbody>
</table>

NOTE

RECOVER PLUS provides migration checking when the INCOPY option is used, and performs header checking and provides messages (BMC40470, BMC40471, and BMC40472) to indicate incompatible migrations. Some examples of incompatible migrations are:

- migration of a partitioned table space into a nonpartitioned table space
- migration of a segmented table space into a table space with a different segment size
- migration into a table space with a different page size

See the Backup and Recovery Products for DB2 Messages Manual for message descriptions.

Example JCL for this process is shown in Figure 144 on page 516.

NOTE

To illustrate that you do not need to specify the DBIDs, OBIDs, and PSIDs for a single table table space with INCOPY, this example JCL does not include them.
Example 2: Moving an application to another subsystem with index recovery

Figure 144  Example RECOVER PLUS JCL to migrate data with index recovery (part 1 of 2)

```plaintext
//RECOUIC EXEC PGM=AFRMAIN,REGION=0M,
// PARM='DBAJ,AFRAQDN,NEW,MSGLEVEL(1),,RDB2STAT(YES)'  
//STEPLIB DD DISP=SHR,DSN=productlibraries
// DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPICK DD SYSOUT=*
//ACCTCPY DD DSN=AFR.ACCTCPYA,
// UNIT=SYSDA,SPACE=(CYL,(10,10)),
// DISP=(NEW,CATLG)
//TRACPY DD DSN=AFR.TRANCPYA,
// UNIT=SYSDA,SPACE=(CYL,(10,10)),
// DISP=(NEW,CATLG)
//SUMMCPY DD DSN=AFR.SUMMCPYA,
// UNIT=SYSDA,SPACE=(CYL,(10,10)),
// DISP=(NEW,CATLG)
//HISTCPY DD DSN=AFR.HISTCPYA,
// UNIT=SYSDA,SPACE=(CYL,(10,10)),
// DISP=(NEW,CATLG)
//SYSIN DD *

OPTIONS OUTCOPY ASCODED INDEXLOG YES
RECOVER TABLESPACE AFRAUDIT.AFRACCT
OBIXDLAT RESET
INCOPY
  FULL DSNAMES AFR.ACCTCPY1
  SHRLEVEL REFERENCE
  INVOLUME 31177B
  INDEVT CART
  OUTCOPY YES OUTCOPYDDN(ACTCYPY)
  TOCOPY LASTCOPY
RECOVER TABLESPACE AFRAUDIT.AFRTRAN
OBIXDLAT RESET
INCOPY
  FULL DSNAMES AFR.TRANCPY1
  SHRLEVEL REFERENCE
  INVOLUME 311763
  INDEVT CART
  OUTCOPY YES OUTCOPYDDN(TRANCYPY)
  TOCOPY LASTCOPY
RECOVER TABLESPACE AFRAUDIT.AFRSUMM
OBIXDLAT RESET
INCOPY
  FULL DSNAMES AFR.SUMMCPY1
  SHRLEVEL REFERENCE
  INVOLUME 311733
  INDEVT CART
  OUTCOPY YES OUTCOPYDDN(SUMMCPY)
  TOCOPY LASTCOPY
RECOVER TABLESPACE AFRAUDIT.AFRHIST
OBIXDLAT RESET
INCOPY
  FULL DSNAMES AFR.HISTCPY1
  SHRLEVEL REFERENCE
```
Example 2: Moving an application to another subsystem with index recovery

<table>
<thead>
<tr>
<th>INVOLUME 311722</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDEVT CART</td>
</tr>
<tr>
<td>OUTCOPY YES OUTCOPYDDN(HISTCOPY)</td>
</tr>
<tr>
<td>TOCOPY LASTCOPY</td>
</tr>
<tr>
<td>RECOVER INDEX OWNERID.AFRACCTI</td>
</tr>
<tr>
<td>OBIDXLAT RESET</td>
</tr>
<tr>
<td>INCOPY</td>
</tr>
<tr>
<td>FULL DSNAME AFR.ACCICPY1</td>
</tr>
<tr>
<td>INVOLUME 311744</td>
</tr>
<tr>
<td>INDEVT CART</td>
</tr>
<tr>
<td>TOCOPY LASTCOPY</td>
</tr>
<tr>
<td>RECOVER INDEX OWNERID.AFRTRANI</td>
</tr>
<tr>
<td>OBIDXLAT RESET</td>
</tr>
<tr>
<td>INCOPY</td>
</tr>
<tr>
<td>FULL DSNAME AFR.TRNICPY1</td>
</tr>
<tr>
<td>INVOLUME 311744</td>
</tr>
<tr>
<td>INDEVT CART</td>
</tr>
<tr>
<td>TOCOPY LASTCOPY</td>
</tr>
<tr>
<td>RECOVER INDEX OWNERID.AFRSUMMI</td>
</tr>
<tr>
<td>OBIDXLAT RESET</td>
</tr>
<tr>
<td>INCOPY</td>
</tr>
<tr>
<td>FULL DSNAME AFR.SUMICPY1</td>
</tr>
<tr>
<td>INVOLUME 311744</td>
</tr>
<tr>
<td>INDEVT CART</td>
</tr>
<tr>
<td>TOCOPY LASTCOPY</td>
</tr>
<tr>
<td>RECOVER INDEX OWNERID.AFRHISTI</td>
</tr>
<tr>
<td>OBIDXLAT RESET</td>
</tr>
<tr>
<td>INCOPY</td>
</tr>
<tr>
<td>FULL DSNAME AFR.HISICPY1</td>
</tr>
<tr>
<td>INVOLUME 311744</td>
</tr>
<tr>
<td>INDEVT CART</td>
</tr>
<tr>
<td>TOCOPY LASTCOPY</td>
</tr>
</tbody>
</table>

**NOTE**

For an index, if you code an OBID clause, you must code two OBID clauses, one for the index and one for the table on which the index is built. When INCOPY is specified for an index, you may omit both OBID clauses, in which case the source OBID for the index and the table is taken from the image copy. If you do not omit the ID clauses, you must code both the target and source values for both OBID clauses.

8 Allow processing on the audit system.

The audit system is now ready for use.
Example 3: Moving an application to another subsystem with index rebuilding

Your new assignment is to build a shadow of an application that runs on a non-data sharing system for use in auditing. At the target site, some new indexes are created to facilitate the audit. The audit application begins at the end of the year.

The table spaces and index spaces for the shadow application are on another DB2 system that is on another computer system with no shared DASD.

**NOTE**
This example uses REBUILD INDEX, which allows you to have *different source and target indexes*.

You can also use this technique when you want to create a different set of indexes for your target spaces on the same DB2 system or at the same site.

1 Obtain an RBA for a consistent view of the source table spaces.

The table spaces for the application are **AFRACCT**, **AFRTRAN**, **AFRSUMM**, and **AFRHIST**.

The table spaces are in database **AFRFIN**. They need to be at a consistent point for the migration. The point at which logging starts after DB2 goes down normally, flushes all buffers, handles all transactions, and comes up again can be used for a point of consistency. A sample of DB2 logging messages is shown in Figure 145.

---

**Figure 145  Sample messages from DB2 logging**

```
DSNR004I > RESTART...UR STATUS COUNTS
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
DSNR005I > RESTART...COUNTS AFTER FORWARD RECOVERY
IN COMMIT=0, INDOUBT=0
DSNR006I > RESTART...COUNTS AFTER BACKWARD RECOVERY
.
.
.
DSNJ099I > LOG RECORDING TO COMMENCE WITH
STARTRBA=0004D176CD5C
```
Example 3: Moving an application to another subsystem with index rebuilding

**NOTE**

Rather than having to determine the RBA for the last shutdown, in this example the option TORBA LASTSHUTDOWN is used to allow RECOVER PLUS to select the last shutdown RBA.

This technique is used because the system is taken down for year-end synchronization. Another technique for establishing a point of consistency is to issue a QUIESCE command for all of the table spaces in one statement. (See “Example 1: Creating a consistent database for query and review” on page 501.) You could also use the point of consistency from an ARCHIVE LOG MODE (QUIESCE) command.

2 Create migration images.

Run RECOVER PLUS to create migration images. The syntax is completely described in Chapter 3, “RECOVER PLUS syntax.”

In the example JCL shown in Figure 146, you create migration images of the table spaces at the point of the pause of the DB2 subsystem that you captured step 1 on page 518. These data sets are identical in format to an image copy or DSN1COPY of the spaces. However, you do not register these copies because they are going to be shipped to the audit system. Also, because they are going to a different computer system, you do not catalog these copies with the operating system.

If you want to make registered copies, see Chapter 9, “Making copies from the log and earlier copies.”

Note that AFRACCT is partitioned, but the OUTCOPY ASCODED option is used to make a copy of all of the partitions.

**Figure 146  Example RECOVER PLUS JCL to create migration images (part 1 of 2)**

```
//RECOUTC1 EXEC PGM=AFRMAIN,REGION=0M,
  // PARM="DBAN,AFRAQDN,NEW,MSGLEVEL(1)"
  // STEPLIB DD DISP=SHR,DSN=product.libraries
  // DD DISP=SHR,DSN=DB2.DSNEXIT
  // DD DISP=SHR,DSN=DB2.DSNLOAD
  // SYSPICK DD SYSOUT=* 
  // ACCTCPY DD DSN=AFR.ACCTCPY1, 
  // UNIT=CART, 
  // DISP=(NEW,KEEP) 
  // TRANCPY DD DSN=AFR.TRANCPY1, 
  // UNIT=CART, 
  // DISP=(NEW,KEEP) 
  // SUMMCPY DD DSN=AFR.SUMMCPY1, 
  // UNIT=CART, 
  // DISP=(NEW,KEEP) 
  // HISTCPY DD DSN=AFR.HISTCPY1, 
  // UNIT=CART, 
  // DISP=(NEW,KEEP) 
  // SYSIN DD * 
```
Example 3: Moving an application to another subsystem with index rebuilding

Obtain data definitions.

Obtain the definitions of the database, table spaces, and tables for which you want to create the audit application for querying. These are the source objects. Also, obtain definitions for the indexes if you want to create these in the audit application.

**NOTE**

See “Defining target objects for migration” on page 500 for more information.

You cannot change any of the characteristics of the table space or tables that affect their internal structure. The AFRTRAN table space must remain partitioned and use the same key. You may change or add to the other indexes used because you are rebuilding them on the target DB2.

For this example, you would obtain the DDL shown in Figure 147.

**Figure 147  Data definitions for the source objects (part 1 of 2)**

<table>
<thead>
<tr>
<th>SQL Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE DATABASE AFRFIN;</td>
<td></td>
</tr>
<tr>
<td>CREATE TABLESPACE AFRTRAN IN AFRFIN USING STOGROUP AFRTEST;</td>
<td></td>
</tr>
<tr>
<td>CREATE TABLESPACE AFRHIST IN AFRFIN USING STOGROUP AFRTEST;</td>
<td></td>
</tr>
<tr>
<td>CREATE TABLE AFRTRAN (ACCTNO INT, TRANSDATE DATE)</td>
<td></td>
</tr>
</tbody>
</table>
Example 3: Moving an application to another subsystem with index rebuilding

4 Obtain object IDs for the source objects.

On the source system, obtain the object IDs. You can do this by using SQL statements to query the catalog or by using some other tool. The example SQL statements and the output are shown Figure 148 on page 522.

NOTE

You can use the default values for the source DBID and PSID. If the table space contains only one table, you can also use the default values for the target OBIDs. RECOVER PLUS extracts this information from the image copy.
Example 3: Moving an application to another subsystem with index rebuilding

522
RECOVER PLUS for DB2 Reference Manual

Create objects on the target system.

Create the database, table spaces, and tables on the target system. They need to be consistent in structure with the source system. (RECOVER PLUS provides migration checking when INCOPY is used. See the second note on page 525 for more information.)

The partitioned index needs to be the same, but the other indexes can be deleted or changed. You can also create new indexes. In this example, you add a new index, AFRTRANX, as shown in Figure 149.

Figure 148  Example SQL and output to obtain object IDs for source objects

```
SELECT NAME, DBID, PSID FROM SYSIBM.SYSTABLESPACE WHERE
  NAME = 'AFRTRAN' OR NAME = 'AFRACCT' OR
  NAME = 'AFRHIST' OR NAME = 'AFRSUMM';
```

```
+----------------------------------+
|   NAME   |   DBID    |   PSID    |
+----------------------------------+
| AFRACCT  |       321 |         2 |
| AFRHIST  |       321 |         8 |
| AFRSUMM  |       321 |         6 |
| AFRTRAN  |       321 |         4 |
+----------------------------------+

SELECT NAME, OBID FROM SYSIBM.SYSTABLES WHERE DBNAME = 'AFRFIN';

```

```
+----------------------------------+
|   NAME   |   OBID    |
+----------------------------------+
| AFRACCT  |         9 |
| AFRTRAN  |        10 |
| AFRSUMM  |        11 |
| AFRHIST  |        12 |
+----------------------------------+
```

Figure 149  Data definitions for target objects (part 1 of 2)

```
CREATE DATABASE AFRAUDIT;
CREATE TABLESPACE AFRHOLD1 IN AFRAUDIT USING STOGROUP AFRTEST;
CREATE TABLESPACE AFRHOLD2 IN AFRAUDIT USING STOGROUP AFRTEST;
CREATE TABLESPACE AFRACCT  IN AFRAUDIT USING STOGROUP AFRTEST NUMPARTS 3;
CREATE TABLESPACE AFRTRAN  IN AFRAUDIT USING STOGROUP AFRTEST;
CREATE TABLESPACE AFRSUMM IN AFRAUDIT USING STOGROUP AFRTEST;
CREATE TABLESPACE AFRHIST  IN AFRAUDIT USING STOGROUP AFRTEST;
CREATE TABLE AFRACCT (  
```
Example 3: Moving an application to another subsystem with index rebuilding

### Figure 149 Data definitions for target objects (part 2 of 2)

```sql
CREATE TABLE AFRACCT
(
    ACCTNO INT,
    ACCTDESC CHAR(30),
    ACCTTYPE CHAR(10),
    BALANCE DECIMAL (10,2)
) IN AFRAUDIT.AFRACCT;

CREATE TABLE AFRTRAN
(
    ACCTNO INT,
    TRANSDATE DATE,
    TRANSDESC CHAR(40),
    CLERKID CHAR(4),
    TOTAL DECIMAL (10,2)
) IN AFRAUDIT.AFRTRAN;

CREATE TABLE AFRSUMM
(
    ACCTNO INT,
    REPCAT CHAR (40),
    SUMMCAT CHAR(40),
    SUMMDATE DATE,
    TOTAL DECIMAL (10,2)
) IN AFRAUDIT.AFRSUMM;

CREATE TABLE AFRHIST
(
    ACCTNO INT,
    TRANSDATE DATE,
    TRANSDESC CHAR(40),
    CLERKID CHAR(4),
    TOTAL DECIMAL (10,2)
) IN AFRAUDIT.AFRHIST;

CREATE INDEX AFRACCTI ON AFRACCT (ACCTNO)
USING STOGROUP AFRTEST
CLUSTER
(
    PART 1 VALUES(30000),
    PART 2 VALUES(60000),
    PART 3 VALUES(99999)
);

CREATE UNIQUE INDEX AFRTRANI ON AFRTRAN (ACCTNO)
USING STOGROUP AFRTEST;
CREATE UNIQUE INDEX AFRTRANX ON AFRTRAN (CLERKID)
USING STOGROUP AFRTEST;
CREATE UNIQUE INDEX AFRSUMMI ON AFRSUMM
(ACCTNO,REPCAT,SUMMCAT,SUMMDATE)
USING STOGROUP AFRTEST;
CREATE INDEX AFRHISTI ON AFRHIST (ACCTNO)
USING STOGROUP AFRTEST;
```
6 Obtain IDs on the target system and set up translation.

On the target system, obtain the object IDs that you need for translation. You can do this by using SQL statements to query the catalog. The example SQL statements and output are shown in Figure 150.

---

**NOTE**

You can use the default values for the target DBID and PSID. If the table space contains only one table, you can also use the default values for the target OBIDs. RECOVER PLUS extracts this information from the catalog.

---

**Figure 150  Example SQL and output to obtain object IDs for target objects**

```sql
SELECT NAME, DBID, PSID FROM SYSIBM.SYSTABLESPACE WHERE NAME = 'AFRTRAN' OR NAME = 'AFRACCT' OR NAME = 'AFRHIST' OR NAME = 'AFRSUMM';
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>DBID</th>
<th>PSID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. AFRACCT</td>
<td>336</td>
<td>6</td>
</tr>
<tr>
<td>2. AFRHIST</td>
<td>336</td>
<td>12</td>
</tr>
<tr>
<td>3. AFRSUMM</td>
<td>336</td>
<td>10</td>
</tr>
<tr>
<td>4. AFRTRAN</td>
<td>336</td>
<td>8</td>
</tr>
</tbody>
</table>

```sql
SELECT NAME, OBID FROM SYSIBM.SYSTABLES WHERE DBNAME = 'AFRAUDIT';
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>OBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. AFRACCT</td>
<td>13</td>
</tr>
<tr>
<td>2. AFRTRAN</td>
<td>14</td>
</tr>
<tr>
<td>3. AFRSUMM</td>
<td>15</td>
</tr>
<tr>
<td>4. AFRHIST</td>
<td>16</td>
</tr>
</tbody>
</table>

---

7 Recover with INCOPY and ID translation.

The tapes from the source system are delivered at the target system. You use the external label information to complete the JCL to create data for the target with RECOVER PLUS. The ID translation is built from the data in step 4 on page 521 and step 6. Output copies are made to allow a normal recovery on the audit system, if required.
Example 3: Moving an application to another subsystem with index rebuilding

**NOTE**

RECOVER PLUS provides migration checking when the INCOPY option is used, and performs header checking and provides messages (BMC40470, BMC40471, and BMC40472) to indicate incompatible migrations. Some examples of incompatible migrations are:

- migration of a partitioned table space into a nonpartitioned table space
- migration of a segmented table space into a table space with a different segment size
- migration into a table space with a different page size

See the *Backup and Recovery Products for DB2 Messages Manual* for message descriptions.

Example JCL for this process is shown in Figure 151.

**Figure 151 Example RECOVER PLUS JCL to migrate data with index rebuilding (part 1 of 2)**

```bash
//RECOVER EXEC PGM=AFRMAIN,REGION=OM,
// PARM='DBAJ,AFRAQDN.NEW,MSGLEVEL(1),.RDB2STAT(YES)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN DD DSN=AFR.ACCTCPYA,
// UNIT=SYSDA,SPACE=(CYL,(10,10)),
// DISP=(NEW,CATLG)
//TRANCPY DD DSN=AFR.TRANCPYA,
// UNIT=SYSDA,SPACE=(CYL,(10,10)),
// DISP=(NEW,CATLG)
//SUMMCPY DD DSN=AFR.SUMMCPYA,
// UNIT=SYSDA,SPACE=(CYL,(10,10)),
// DISP=(NEW,CATLG)
//HISTCPY DD DSN=AFR.HISTCPYA,
// UNIT=SYSDA,SPACE=(CYL,(10,10)),
// DISP=(NEW,CATLG)
//SORTOUT DD DUMMY
//SYSIN DD *
//OPTIONS OUTCOPY ASCODED
RECOVER TABLESPACE AFRAUDIT.AFRACCT
    OBIDXLAT RESET DBID(321,336) PSID(2,6)
    OBID(9,13)
    INCOPY
        FULL DSNNAME AFR.ACCTCPY
        SHRLEVEL REFERENCE
        INVOLUME 311778
        INDEVT CART
        OUTCOPY YES OUTCOPYDDN(ACCTCPY)
        TOCOPY LASTCOPY
RECOVER TABLESPACE AFRAUDIT.AFRTRAN
    OBIDXLAT RESET DBID(321,336) PSID(4,8)
    OBID(10,14)
    INCOPY
        FULL DSNNAME AFR.TRANCPY
```
Example 3: Moving an application to another subsystem with index rebuilding

Figure 151  Example RECOVER PLUS JCL to migrate data with index rebuilding (part 2 of 2)

<table>
<thead>
<tr>
<th>SHRLLEVEL REFERENCE</th>
<th>INVOLUME 311763</th>
<th>INDEVT CART</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTCOPY YES OUTCOPYDDN(TRANCPY)</td>
<td>TOCOPY LASTCOPY</td>
<td></td>
</tr>
<tr>
<td>RECOVER TABLESPACE AFRAUDIT.AFRSUMM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OBIDXLAT RESET DBID(321,336)  PSID(6,10)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OBID(11,15)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INCOPY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FULL DSNAME AFR.SUMMCPY1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHRLLEVEL REFERENCE</td>
<td>INVOLUME 311733</td>
<td>INDEVT CART</td>
</tr>
<tr>
<td>OUTCOPY YES OUTCOPYDDN(SUMMCPY)</td>
<td>TOCOPY LASTCOPY</td>
<td></td>
</tr>
<tr>
<td>RECOVER TABLESPACE AFRAUDIT.AFRHIST</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OBIDXLAT RESET DBID(321,336)  PSID(8,12)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OBID(12,16)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INCOPY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FULL DSNAME AFR.HISTCPY1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHRLLEVEL REFERENCE</td>
<td>INVOLUME 311722</td>
<td>INDEVT CART</td>
</tr>
<tr>
<td>OUTCOPY YES OUTCOPYDDN(HISTCPY)</td>
<td>TOCOPY LASTCOPY</td>
<td></td>
</tr>
<tr>
<td>REBUILD INDEX(ALL)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TABLESPACE AFRAUDIT.AFRACCT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOWORKDDN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REBUILD INDEX(ALL)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TABLESPACE AFRAUDIT.AFRTRAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOWORKDDN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REBUILD INDEX(ALL)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TABLESPACE AFRAUDIT.AFRSUMM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOWORKDDN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REBUILD INDEX(ALL)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TABLESPACE AFRAUDIT.AFRHIST</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOWORKDDN</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8 Allow processing on the audit system.

The audit system is now ready for use.
Making copies from the log and earlier copies

This chapter presents the following topics:

Overview .......................................................... 527
Example 1: Creating a consistent set of copies at a point in time ............. 529
Example 2: Creating an image copy without accessing the space ............ 532

Overview

This chapter provides examples of using the RECOVER PLUS product to create image copies.

You can use RECOVER PLUS to make updated image copies from typical recovery resources such as earlier copies, change accumulation files, and DB2 logs. Figure 152 on page 528 shows RECOVER PLUS using a full copy, multiple incremental copies, a change accumulation file (created by using the R+/CHANGE ACCUM product), and DB2 log files to produce an updated full image copy.
During an OUTCOPY ONLY operation, RECOVER PLUS writes the output to a sequential image copy data set instead of a DB2 space. This process allows you to make copies without accessing the DB2 space or interfering with normal DB2 access in any way. If you elect to end the process at any of the following points, RECOVER PLUS registers the new copy, if it is registered, as a SHRLEVEL REFERENCE full image copy:

- the log point of a SHRLEVEL REFERENCE incremental copy
- the log point of a quiesce point
- the log point of the last -ARCHIVE MODE(QUIESCE) command
- the last successful subsystem shutdown (in a non–data sharing environment)

In all other cases, RECOVER PLUS registers the new copy as a SHRLEVEL CHANGE copy.
Example 1: Creating a consistent set of copies at a point in time

You suspect a table space or index has been damaged by an application or a media failure. If you have a point of consistency but no copy close to the point of the damage, you can use this procedure to create a copy while the investigation of the damage is still underway. If the spaces must be recovered, you will have a single copy created for each space that allows a recovery without requiring log or incremental copies.

1 Find or create a point of consistency.

The most common way to achieve a point of consistency is to issue a QUIESCE command on a set of table spaces in a single statement as shown in Figure 153.

Figure 153 Example JCL to QUIESCE a set of table spaces

```
//QUIESCE1 EXEC DSNUPROC,SYSTEM=DBAJ,UID='AFRRM31'.
// UTPROC='',REGION=4M,
// LIB='SYS2.DBA.ZD.PROD.DSNLOAD'
//DSNUPROC.SYSPRINT DD SYSOUT=* 
//DSNUPROC.SYSDISC DD DUMMY 
//DSNUPROC.UTPRINT DD SYSOUT=* 
//DSNUPROC.SYSERR DD UNIT=WORK,SPACE=(CYL,(2,2)) 
//DSNUPROC.SYSMAP DD UNIT=WORK,SPACE=(CYL,(2,2)) 
//DSNUPROC.SYSIN DD *
QUIESCE TABLESPACE AFRFIN.AFRACCT
   TABLESPACE AFRFIN.AFRTRAN
   TABLESPACE AFRFIN.AFRSUMM
   TABLESPACE AFRFIN.AFRHIST
```

Output from the QUIESCE follows in Figure 154.

Figure 154 Example QUIESCE output

```
DSNU050I DSNUGUTC - QUIESCE TABLESPACE AFRFIN.AFRACCT TABLESPACE AFRFIN.AFRTRAN TABLESPACE AFRFIN.AFRSUMM TABLESPACE AFRFIN.AFRHIST
DSNU477I > DSNUQUIA - QUIESCE SUCCESSFUL FOR TABLESPACE AFRFIN.AFRACCT
DSNU477I > DSNUQUIA - QUIESCE SUCCESSFUL FOR TABLESPACE AFRFIN.AFRTRAN
DSNU477I > DSNUQUIA - QUIESCE SUCCESSFUL FOR TABLESPACE AFRFIN.AFRSUMM
DSNU477I > DSNUQUIA - QUIESCE SUCCESSFUL FOR TABLESPACE AFRFIN.AFRHIST
DSNU474I > DSNUQUIB - QUIESCE AT RBA 0004D176C05C
DSNU475I > DSNUQUIB - QUIESCE UTILITY COMPLETE, ELAPSED TIME= 00:00:00
DSNU010I DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
```
Example 1: Creating a consistent set of copies at a point in time

Other ways to obtain a point of consistency include using

- the point of a SHRLEVEL REFERENCE incremental copy
- the point of an ARCHIVE LOG MODE(QUIESCE)

You may use the keyword TOLOGPOINT LASTARCHQ.

- a DB2 shutdown with spaces in a consistent state

You may use the keyword TOLOGPOINT LASTSHUTDOWN.

- the start of the first member of a data sharing group to be started, assuming the spaces were in a consistent state and all members were down

- RECOVERY MANAGER for DB2 to locate quiet points based on log ranges

- Log Master to locate quiet points in the log

2 Recover with OUTCOPY ONLY using TORBA (or TOLOGPOINT).

The point of consistency, which in this case is the quiesce point indicated by TORBA LASTQUIESCE, is now used to create new copies from the old copies and logs without reading the space itself.

The example Figure 155 uses

- the dynamic allocation of output data sets with wildcards for defining the data set name
- the automatic generation of GDGs using the AFRGDG DD statement
- the recovery of all indexes associated with each table space that is being recovered

Figure 155  Example RECOVER PLUS JCL to create copies using dynamic allocation with automatic generation of GDGs (part 1 of 2)

```plaintext
//RECOVER EXEC PGM=AFRMAIN,REGION=0M,
// PARM='DBAJ.AFRAQDN.NEW,MSGLEVEL(1)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// DD DISP=SHR,DSN=DB2.DSNLOAD
//AFRGDG DD DSN=AFR.PROD.GDGDEF(GDGDEF01),DISP=SHR
//SYSIN DD *
//OPTIONS OUTCOPY ASCODED INDEXLOG YES

OUTPUT COPY DSNAME AFR.IC.&DB.&SP(+1) UNIT CART CATLG YES STACK YES
```
Example 1: Creating a consistent set of copies at a point in time

Figure 155  Example RECOVER PLUS JCL to create copies using dynamic allocation with automatic generation of GDGs (part 2 of 2)

<table>
<thead>
<tr>
<th>RECOVER TABLESPACE AFRIN.AFRACCT</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY</td>
</tr>
<tr>
<td>INDEX ALL TABLESPACE AFRIN.AFRACCT</td>
</tr>
<tr>
<td>OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY</td>
</tr>
<tr>
<td>TABLESPACE AFRIN.AFRTRAN</td>
</tr>
<tr>
<td>OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY</td>
</tr>
<tr>
<td>INDEX ALL TABLESPACE AFRIN.AFRTRAN</td>
</tr>
<tr>
<td>OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY</td>
</tr>
<tr>
<td>TABLESPACE AFRIN.AFRSUMM</td>
</tr>
<tr>
<td>OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY</td>
</tr>
<tr>
<td>INDEX ALL TABLESPACE AFRIN.AFRSUMM</td>
</tr>
<tr>
<td>OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY</td>
</tr>
<tr>
<td>TABLESPACE AFRIN.AFRHIST</td>
</tr>
<tr>
<td>OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY</td>
</tr>
<tr>
<td>INDEX ALL TABLESPACE AFRIN.AFRHIST</td>
</tr>
<tr>
<td>OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY</td>
</tr>
<tr>
<td>TORBA LASTQUIESCE</td>
</tr>
</tbody>
</table>
Example 2: Creating an image copy without accessing the space

You can create a new image copy of a table space or index at any time by running a RECOVER command with OUTCOPY ONLY. The underlying space will not be accessed or affected in any way.

- If you do not specify a log point (with TORBA or TOLOGPOINT), the log is used up to the point where the utility begins. The log information is combined with any copies or change accumulation files to create a new full copy.

- If you do specify a log point (with TORBA or TOLOGPOINT), the log is used up to the target LRSN. The log information is combined with any copies or change accumulation files to create a new full copy.

Whether or not you select a TORBA or TOLOGPOINT, RECOVER PLUS registers the copy as SHRLEVEL CHANGE unless the log point is a point of consistency.

Example JCL to create the new image copy and use dynamic allocation for the output image copy is shown in Figure 156.

---

**Figure 156** Example RECOVER PLUS JCL to create new image copy using dynamic allocation (part 1 of 2)

```bash
//RECOVER EXEC PGM=AFRMAIN,REGION=0M, // PARM=’DBAN,AFRAQDN,NEW,MSGLEVEL(1)’ //STEPLIB DD DISP=SHR,DSN=product.libraries // DD DISP=SHR,DSN=DB2.DSNEXIT // DD DISP=SHR,DSN=DB2.DSNLOAD //AFRGDG DD DSN=AFR.PROD.GDGDEF(GDGDEF01),DISP=SHR //SYSIN DD * OPTIONS OUTCOPY ASCODED INDEXLOG YES OUTPUT COPY DSNNAME AFR.IC.&DB.&SP(+1) UNIT SYSDA CATLG YESoodles MODELDCB SYS1.MODEL SPACE (10,10) CYL RECOVER TABLESPACE AFRFIN.AFRACCT OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY TABLESPACE AFRFIN.AFRTRAN OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY TABLESPACE AFRFIN.AFRSUMM OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY TABLESPACE AFRFIN.AFRHIST OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY INDEX AFRFIN.AFRACCTI OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY INDEX AFRFIN.AFRTRANI OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY INDEX AFRFIN.AFRSUMMI OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY INDEX AFRFIN.AFRSUMMI OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY
```
Figure 156  Example RECOVER PLUS JCL to create new image copy using dynamic allocation (part 2 of 2)

```
INDEX AFRFIN.AFRHISTI
OUTCOPY ONLY REGISTER ALL  OUTCOPYDDN COPY
```
Example 2: Creating an image copy without accessing the space

RECOVER PLUS for DB2 Reference Manual
RECOVER PLUS installation options

This appendix presents the following topics.

Overview ......................................................... 535
Installation options macro listing ............................ 536
RECOVER PLUS installation options .......................... 537

Overview

The RECOVER PLUS for DB2 product is installed by using the Installation System from BMC. During the installation, the customization process generates a customized installation data set. This data set contains customized jobs that install RECOVER PLUS into your specific DB2® environment. One of these jobs, $C30DOPT, establishes the default processing option values that RECOVER PLUS uses.

The $C30DOPT job assembles the options macro. The macro contains the RECOVER PLUS processing options and the values for those options that are shipped with RECOVER PLUS. When the Installation System-generated customization job is submitted, it links the AFRS$OPTS installation options module in the APF-authorized library that is designated by your site. If any values for these options are changed during customization, the new values override the values from the options macro.

You can customize the installation of RECOVER PLUS by changing the values for the RECOVER PLUS installation options. However, if you change any of the values in $C30DOPT after RECOVER PLUS has been installed, you must rerun the jobs for these changes to take effect.

You can also create additional options modules that allow you to use different values of these options for different executions of RECOVER PLUS. For example, you might use the default installation options module for most jobs but create another options module with customized values for certain options for special situations. For
information about specifying an options module at runtime, see Chapter 4, “Building and executing RECOVER PLUS jobs.” For more information about customizing your installation of RECOVER PLUS, see the Backup and Recovery Products for DB2 Installation Guide or the Recovery Management for DB2 Installation Guide.

## Installation options macro listing

Figure 157 shows the installation options macro listing for RECOVER PLUS.

### Figure 157   RECOVER PLUS installation options module

<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$AFROPTS</td>
<td>&amp;TBUFFS=1000,</td>
<td>BUFFER MANAGER PAGES</td>
</tr>
<tr>
<td>&amp;PLANRECV</td>
<td>=AFRBvvr,</td>
<td>RECOVER PLUS MAIN PLAN NAME</td>
</tr>
<tr>
<td>&amp;OPNDB2ID</td>
<td>=YES,</td>
<td>USE DB2 RACF ID</td>
</tr>
<tr>
<td>&amp;CHECKPT</td>
<td>=PHASE,</td>
<td>CHECKPOINT FOR RESTART</td>
</tr>
<tr>
<td>&amp;CHECKINT</td>
<td>=0,</td>
<td>INTERVAL BETWEEN CHECKPOINTS (MIN)</td>
</tr>
<tr>
<td>&amp;RDB2STAT</td>
<td>=YES,</td>
<td>RESTORE ORIGNAL STATUS OF SPACES</td>
</tr>
<tr>
<td>&amp;RCLTSK</td>
<td>=10,</td>
<td>MAXIMUM RECALL TASKS</td>
</tr>
<tr>
<td>&amp;AMSCAT</td>
<td>=NO,</td>
<td>USE CATALOG OPTION IN IDCAMS DEFINE</td>
</tr>
<tr>
<td>&amp;INDEXLOG</td>
<td>=NO,</td>
<td>SPECIFY RECOVER INDEX BEHAVIOR</td>
</tr>
<tr>
<td>&amp;IXRECP</td>
<td>=NO,</td>
<td>PUT INDEXES IN RECP STATUS if TS PIT RCVR</td>
</tr>
<tr>
<td>&amp;OUTCOPY</td>
<td>=ASCODED,</td>
<td>SPECIFY HOW TO CREATE OUTPUT IMAGE COPIES</td>
</tr>
<tr>
<td>&amp;SORTDEVT</td>
<td>,</td>
<td>DEVICE TYPE FOR SORT WORK SPACE</td>
</tr>
<tr>
<td>&amp;SORTNUM</td>
<td>,</td>
<td>NUMBER OF ALLOCATED SORT WORK SPACE DSNS</td>
</tr>
<tr>
<td>&amp;MAXKSORT</td>
<td>,</td>
<td>MAXIMUM NUMBER OF INDEX KEY SORTS</td>
</tr>
<tr>
<td>&amp;MAXLOGS</td>
<td>=5,</td>
<td>MAXIMUM LOGS TO ALLOCATE</td>
</tr>
<tr>
<td>&amp;MAXDRIVE</td>
<td>,</td>
<td>MAXIMUM TAPE DRIVES TO BE USED</td>
</tr>
<tr>
<td>&amp;SMCORE</td>
<td>=(0K,0K),</td>
<td>MEMORY FOR SORT</td>
</tr>
<tr>
<td>&amp;RESINV</td>
<td>=OK,</td>
<td>MEMORY RESERVED FOR IDCAMS</td>
</tr>
<tr>
<td>&amp;WKUNIT</td>
<td>=SYSSALLDA,</td>
<td>UNIT TO DYNAMICALLY ALLOCATE WORK FILES</td>
</tr>
<tr>
<td>&amp;XBMID</td>
<td>,</td>
<td>XBM SUBSYSTEM ID OR XBM GROUP NAME</td>
</tr>
<tr>
<td>&amp;WTOR</td>
<td>=YES,</td>
<td>WTOR FOR SPACES IN STOPP STATUS</td>
</tr>
<tr>
<td>&amp;ERRCONT</td>
<td>=10,</td>
<td>MAXIMUM TIMES TO CONT ON SEVERE ERROR</td>
</tr>
<tr>
<td>&amp;TZRULE</td>
<td>=NONE,</td>
<td>DAYLIGHT SAVINGS TIME (DST) SETTING</td>
</tr>
<tr>
<td>&amp;LOCCPSEL</td>
<td>=(FC,LP,LB),</td>
<td>LOCAL SITE COPY PRIMARY, SECONDARY</td>
</tr>
<tr>
<td>&amp;LOCCASEL</td>
<td>=(LP,LB),</td>
<td>LOCAL SITE CHANGE ACCUM PRIMARY,SECONDARY</td>
</tr>
<tr>
<td>&amp;REMCPSEL</td>
<td>=(RP,RB,FC),</td>
<td>REMOTE SITE COPY PRIMARY, SECONDARY</td>
</tr>
<tr>
<td>&amp;REMCASEL</td>
<td>=(RP,RB),</td>
<td>REMOTE SITE CHANGE ACCUM PRIMARY,SECONDARYYX</td>
</tr>
<tr>
<td>&amp;KEYDSNAM</td>
<td>,</td>
<td>NAME OF KEY DATA SET FOR ENCRYPTION</td>
</tr>
<tr>
<td>&amp;ACFORTSS</td>
<td>=YES,</td>
<td>USE CA ACF2 OR TOP SECRET SECURITY</td>
</tr>
<tr>
<td>&amp;BINDQUALIER</td>
<td>=AFRvvr,</td>
<td>RECOVER PLUS BIND QUALIFIER</td>
</tr>
<tr>
<td>&amp;PUBLICPLAN</td>
<td>=YES,</td>
<td>RUN RECOVER PLUS WITH PUBLIC PRIVILEGE</td>
</tr>
<tr>
<td>&amp;DATAMVR</td>
<td>,</td>
<td>PROGRAM TO USE FOR NONSNAPPABLE DASD</td>
</tr>
<tr>
<td>&amp;HISTORY</td>
<td>=YES,</td>
<td>BMC HISTORY TABLE</td>
</tr>
<tr>
<td>&amp;MAXLSORT</td>
<td>=0,</td>
<td>NUMBER OF PARALLEL Merges AND LOG SORTS</td>
</tr>
<tr>
<td>&amp;KSORTSHARE</td>
<td>=YES,</td>
<td>SHARE KEY SORTS</td>
</tr>
<tr>
<td>&amp;AUX</td>
<td>=NO,</td>
<td>INCLUDE AUXILIARY OBJECTS IN RECOVERY</td>
</tr>
<tr>
<td>&amp;BACKOUT</td>
<td>=AUTO,</td>
<td>BACKOUT RECOVERY OPTION FOR PIT RECOVERY</td>
</tr>
<tr>
<td>&amp;USEHDROBIDS</td>
<td>=YES,</td>
<td>USE OBIDS IN HEADER WHEN MIGRATING</td>
</tr>
</tbody>
</table>
RECOVER PLUS installation options

This section describes the RECOVER PLUS installation options including their default values.

For quick reference, Table 19 presents the options in alphabetical order and includes their default values, a brief description, and a reference to more information.

Table 19  RECOVER PLUS installation options  (part 1 of 3)

<table>
<thead>
<tr>
<th>RECOVER PLUS installation option</th>
<th>Default valuea</th>
<th>Brief description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACFORTSS</td>
<td>YES</td>
<td>specifies use of CA ACF2 security or CA Top Secret security</td>
<td>552</td>
</tr>
<tr>
<td>AMSCAT</td>
<td>NO</td>
<td>defines use of catalog option in the IDCAMS define</td>
<td>541</td>
</tr>
<tr>
<td>AUTOSIZE</td>
<td>YES</td>
<td>specifies dynamic sizing for change accumulation files</td>
<td>551</td>
</tr>
<tr>
<td>AUX</td>
<td>NO</td>
<td>specifies if the auxiliary objects related to a base table space are to be included in recovery, and for DB2 Version 10 and later, if the history table related to a specified system-maintained temporal table is included in the recovery</td>
<td>555</td>
</tr>
<tr>
<td>BACKOUT</td>
<td>AUTO</td>
<td>specifies the backout recovery option</td>
<td>556</td>
</tr>
<tr>
<td>BINDQUALIFIER</td>
<td>AFRvvr</td>
<td>RECOVER PLUS bind qualifier, for example AFR101</td>
<td>553</td>
</tr>
<tr>
<td>CHECKINT</td>
<td>0</td>
<td>interval between checkpoints in minutes</td>
<td>540</td>
</tr>
<tr>
<td>CHECKPT</td>
<td>PHASE</td>
<td>checkpoint for restart</td>
<td>540</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>none</td>
<td>SMS data class</td>
<td>559</td>
</tr>
<tr>
<td>DATAMVR</td>
<td>none</td>
<td>provides XBM with the name of the program to use to copy a data set if the data set is not on snappable DASD</td>
<td>552</td>
</tr>
<tr>
<td>EATTR</td>
<td>none</td>
<td>enable extended attributes to allocate an extended format sequential data set (supported by z/OS Version 1.11 and later)</td>
<td>557</td>
</tr>
<tr>
<td>ERRCONT</td>
<td>10</td>
<td>maximum severe errors</td>
<td>548</td>
</tr>
<tr>
<td>HISTORY</td>
<td>YES</td>
<td>use BMC HISTORY table</td>
<td>553</td>
</tr>
<tr>
<td>INDEXLOG</td>
<td>NO</td>
<td>specifies RECOVER INDEX behavior</td>
<td>541</td>
</tr>
<tr>
<td>IXRECP</td>
<td>NO</td>
<td>puts indexes in RECP status if TS PIT recovery</td>
<td>542</td>
</tr>
<tr>
<td>KEYDSNAM</td>
<td>none</td>
<td>name of the key data set for encryption</td>
<td>552</td>
</tr>
</tbody>
</table>
## RECOVER PLUS installation options

<table>
<thead>
<tr>
<th>RECOVER PLUS installation option</th>
<th>Default valuea</th>
<th>Brief description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSORTSHARE</td>
<td>YES</td>
<td>specifies whether key sorts are shared, which determines number of active key sorts (with MAXKSORT)</td>
<td>554</td>
</tr>
<tr>
<td>LOCCASEL</td>
<td>(LP, LB)</td>
<td>resource selection for local site change accumulation files</td>
<td>551</td>
</tr>
<tr>
<td>LOCCPSEL</td>
<td>(FC, LP, LB)</td>
<td>resource selection for local site copies</td>
<td>551</td>
</tr>
<tr>
<td>MAXDRIVE</td>
<td>0</td>
<td>specifies maximum tape drives to be used</td>
<td>545</td>
</tr>
<tr>
<td>MAXKSORT</td>
<td>min((2 * the number of CPUs), 12)</td>
<td>specifies the maximum number of index key sorts that can run concurrently</td>
<td>544</td>
</tr>
<tr>
<td>MAXLOGS</td>
<td>5</td>
<td>specifies the maximum logs to allocate</td>
<td>545</td>
</tr>
<tr>
<td>MAXLSORT</td>
<td>0</td>
<td>specifies the maximum number of log sorts that can run concurrently and determines the number of MERGE/RESTORE/SNAP phases that can run in parallel whether or not log records are processed</td>
<td>553</td>
</tr>
<tr>
<td>OPNDB2ID</td>
<td>YES</td>
<td>specifies the use of the DB2 RACF ID</td>
<td>539</td>
</tr>
<tr>
<td>OUTCOPY</td>
<td>ASCODED</td>
<td>specifies how to create output image copies</td>
<td>542</td>
</tr>
<tr>
<td>PLANRECV</td>
<td>AFRBvvr</td>
<td>RECOVER PLUS main plan name, for example AFRBvvr</td>
<td>539</td>
</tr>
<tr>
<td>PUBLICPLAN</td>
<td>YES</td>
<td>grants PUBLIC privilege to run RECOVER PLUS</td>
<td>553</td>
</tr>
<tr>
<td>RCLTSK</td>
<td>10</td>
<td>specifies the maximum recall tasks</td>
<td>540</td>
</tr>
<tr>
<td>RDB2STAT</td>
<td>YES</td>
<td>restores original status of spaces</td>
<td>540</td>
</tr>
<tr>
<td>REMCASEL</td>
<td>(RP, RB)</td>
<td>resource selection for remote site change accumulation files</td>
<td>551</td>
</tr>
<tr>
<td>REMCPSEL</td>
<td>(RP, RB, FC)</td>
<td>resource selection for remote site copies</td>
<td>551</td>
</tr>
<tr>
<td>RESINV</td>
<td>0K</td>
<td>specifies memory reserves for IDCAMS</td>
<td>546</td>
</tr>
<tr>
<td>SCORE</td>
<td>(0K, 0K)</td>
<td>specifies memory for sort</td>
<td>545</td>
</tr>
<tr>
<td>SORTDEVT</td>
<td>determined by BMCSORT installation options</td>
<td>specifies device type for sort work space</td>
<td>543</td>
</tr>
<tr>
<td>SORTNUM</td>
<td>determined by BMCSORT installation options</td>
<td>specifies number of allocated sort work space DSNs</td>
<td>543</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>none</td>
<td>SMS storage class</td>
<td>559</td>
</tr>
<tr>
<td>TBUFFS</td>
<td>1000</td>
<td>specifies buffer manager pages</td>
<td>539</td>
</tr>
<tr>
<td>TZRULE</td>
<td>NONE</td>
<td>specifies the rule that determines when daylight savings time (DST) begins and ends (Use this installation option to enable DST adjustments.)</td>
<td>548</td>
</tr>
</tbody>
</table>
Descriptions of each of the installation options with their default values follow.

**TBUFFS=1000**

This option specifies the number of pages to allocate to the RECOVER PLUS buffer manager. This pool of pages is allocated when the recovery uses random processes. For example, index builds use the buffer manager. BMC recommends that you modify this parameter only after consultation with your RECOVER PLUS technical support analyst.

**PLANRECV=AFRBvv**

This option is the RECOVER PLUS execution plan. The default value is AFRBvv, where vv represents a two-character version number and r represents a one-character release number, such as AFRB101.

**OPNDB2ID=YES**

This option indicates whether to use the DB2 RACF ID instead of the RACF ID of the user running RECOVER PLUS when opening the DB2 data sets, such as table spaces and logs. Valid values are YES and NO. The default value is YES.

OPNDB2ID=YES uses the RACFID of DB2. OPNDB2ID=NO uses the RACFID of the user running RECOVER PLUS. If you specify OPNDB2ID=NO, the user must have the appropriate RACF authority.

If the RACF started procedures table (ICHRIN03) specifies DB2 as a privileged or trusted task and has no user ID associated with the DB2 address space, you cannot use OPNDB2ID to allow RECOVER PLUS to access the DB2 data sets. In this case, the user running RECOVER PLUS must have RACF authority to access the data sets needed for recovery.
RECOVER PLUS installation options

**NOTE**

OPNDB2ID works under data sharing only if all of the RACF IDs for the members of a group are the same. The authorizations for the BSDS and log data sets must also be the same.

For any security system other than RACF, RECOVER PLUS ignores this option and uses the ID of the user running RECOVER PLUS.

**CHECKPT=PHASE**

This option specifies whether restart checkpoints are taken. Valid values are NO and PHASE. The default value is PHASE.

CHECKPT=NO indicates that no checkpoints are taken. CHECKPT=PHASE specifies that checkpoints are taken at the end of each phase. You can override the CHECKPT installation option by using the checkpoint override parameter (see page 273). Also, see the information about the CHECKINT option (which further limits the frequency of taking checkpoints) that follows.

**CHECKINT=0**

This option specifies the time in minutes between checkpoints. This option allows you to balance the frequency of restart points with the overhead incurred when taking restart points. If CHECKPT=PHASE, a restart point is taken only if more minutes than those specified by CHECKINT have elapsed since the last restart point was taken. The default interval is 0 minutes. Use CHECKINT=0 and CHECKPT=PHASE to force a restart checkpoint after every phase. You can override the CHECKINT installation option by using the CHECKINT keyword with the OPTIONS command (see page 117).

**RDB2STAT=YES**

This option specifies whether the original status of the recovered objects is to be restored after a successful recovery. Valid values are YES and NO. The default value is YES. RDB2STAT=NO leaves the spaces stopped. RDB2STAT=YES restores the original status. You can override the RDB2STAT installation option by using the RDB2STAT override parameter (see page 274).

**RCLTSK=10**

This option specifies the maximum number of concurrent tasks that are used to recall archived image copies and log data sets when you use the EARLYRECALL option. (For more information about EARLYRECALL, see page 112.) RCLTSK=0 disables EARLYRECALL. The default value is 10.
AMSCAT=NO

This option indicates whether to include the CATALOG option in IDCAMS DEFINE CLUSTER commands when defining data sets for STOGROUP-defined spaces. The CATALOG option specifies the VCAT name specified in the STOGROUP definition. AMSCAT=NO is the default and does not include the CATALOG option. AMSCAT=YES includes the CATALOG option.

INDEXLOG=NO

This option specifies what behavior RECOVER INDEX has. Valid values are NO, YES, and AUTO.

- INDEXLOG=NO, the default, indicates that RECOVER INDEX and REBUILD INDEX are synonyms and indexes are rebuilt from the keys extracted from the table space.

- INDEXLOG=YES indicates that RECOVER INDEX recovers indexes from copies, the log, or both, while REBUILD INDEX rebuilds indexes from the keys extracted from the table space.

- INDEXLOG=AUTO indicates that RECOVER INDEX first attempts an index recovery from image copies, the log, or both. However, if an image copy for the index does not exist, RECOVER PLUS converts the request to a REBUILD INDEX request.

INDEXLOG=AUTO supports BACKOUT recoveries. However, conversion from RECOVER INDEX to REBUILD INDEX is not supported for following options:

- OUTCOPY: REBUILD INDEX does not include the OUTCOPY option so this request is not eligible for conversion.

- OBIDXLAT: REBUILD INDEX does not include the OBIDXLAT option so this request is not eligible for conversion.

- TOLOGPOINT: If the RECOVER INDEX request includes the TOLOGPOINT option, the request is eligible for conversion to REBUILD INDEX only if the associated table space is also recovered in the same job step.

- TOCOPY: If the RECOVER INDEX request includes the TOCOPY option, the request is eligible for conversion to REBUILD INDEX only if the associated table space is also recovered in the same job step.

- DSNUM: If the RECOVER INDEX command statement was for a specific data set of a nonpartitioned index, the request is not eligible for conversion to REBUILD INDEX.
RECOVER PLUS installation options

**NOTE**

If you plan to make index image copies, you might consider using INDEXLOG=YES or INDEXLOG=AUTO, but realize that you may have to change some RECOVER INDEX command statements to REBUILD INDEX. However, if you have no immediate plans to start making index image copies, you may prefer to leave the default, INDEXLOG=NO, so that you will not have to change syntax.

You can override the INDEXLOG installation option by using the INDEXLOG keyword with the OPTIONS command (see page 126).

**NOTE**

The RECOVER INDEXSPACE command statement always invokes a recovery based on copies and logs, regardless of the INDEXLOG setting. Also, REBUILD INDEX always rebuilds indexes, regardless of the setting for INDEXLOG.

**IXRECP=NO**

IXRECP specifies whether RECOVER PLUS alerts you that indexes have not been recovered. Valid values are NO and YES. IXRECP=YES tells RECOVER PLUS to issue a warning message for each index that has not been recovered with the table space, and to put each such index in RECP status. IXRECP=NO, the default, disables this feature. You can override the IXRECP installation option by using the IXRECP keyword with the OPTIONS command (see page 128).

**NOTE**

Use this option when a point-in-time recovery of a table space is performed. It is important that any indexes on the space be recovered so that index data is synchronized with the data to which it refers.

**OUTCOPY=ASCODED**

This option specifies how to create output image copies on partitioned table spaces and indexes. Valid values are ASCODED and BYPART. If you specify OUTCOPY=ASCODED, the default, and the recovery specifies DSNUM ALL, RECOVER PLUS creates a DSNUM 0 image copy. If a recovery is by DSNUM and you specify BYPART or ASCODED for this option, RECOVER PLUS creates the image copy by DSNUM. You can override the OUTCOPY installation option by using the OUTCOPY keyword with the OPTIONS command (see page 121).
SORTDEVT

This option specifies the default device type for the temporary sort work data sets that BMCSORT uses for sorting log records and index keys. If you do not specify a value, the value is obtained from the BMCSORT installation options. For more information, see the BMCSORT Reference Manual.

You can override the SORTDEVT installation option by setting the SORTDEVT keyword in one of the following commands:

- OPTIONS
- RECOVER TABLESPACE, RECOVER INDEXSPACE, or RECOVER INDEX
- RECOVER UNLOADKEYS
- REBUILD INDEX

For more information, see “General information about SORTDEVT and SORTNUM” on page 263.

SORTNUM

This option specifies the default number of temporary sort work data sets that are allocated dynamically by BMCSORT for sorting log records and index keys.

You can specify an integer value of 1 through 255. If you do not specify a value, the value is obtained from the BMCSORT installation options.

You can override the SORTNUM installation option by setting the SORTNUM keyword on one of the following commands:

- OPTIONS
- RECOVER TABLESPACE, RECOVER INDEXSPACE, or RECOVER INDEX
- RECOVER UNLOADKEYS
- REBUILD INDEX

When you specify this option, BMCSORT dynamically allocates the number of sort work files that it needs for each sort task up to the maximum that is illustrated in the following formula:

\[ \text{maximum dynamically allocated sort work files} = n - \text{preallocated sort work files} \]

If you specify \textit{integer} from 1 through 32, \(n\) equals 32.

If you specify \textit{integer} greater than 32, \(n\) equals integer.
NOTE
Preallocated sort work files include sort work files that are allocated in your JCL.

For more information, see “General information about SORTDEVT and SORTNUM” on page 263.

MAXKSORT

This option specifies the maximum number of index key sorts that can run in parallel. The value of MAXKSORT also determines the concurrency of index rebuilds. Thus, MAXKSORT can be used to improve recovery performance. (The KSORTSHARE option also impacts index recovery. For more information, see “KSORTSHARE=YES” on page 554.)

NOTE
When you rebuild the indexes of a multi-data-set, nonpartitioned table space, the UNLOADs run serially in the main task, but the REBUILDs are multitasked.

Valid values are 1 to 999. The default value is given by the following formula:

minimum((2 * the number of CPUs), 12)

NOTE
Each sort requires about 256 KB of memory below the line. Values for MAXKSORT greater than 12 are not recommended.

You can override the MAXKSORT installation option by using the MAXKSORT parameter with the OPTIONS command (see page 136). For more information, see “Understanding the UNLOADKEYS and BUILDINDEX strategy” on page 423.

When you specify MAXKSORT, the following files are dynamically allocated if you do not code them in JCL:

- SYSOU<nnn>: sort message files

    <nnn> is a number between 1 and the value that is specified for MAXKSORT.
- **SxxxxWKnn**: key sort work files

  `xxx` is a number between 1 and the value that is specified for MAXKSORT. `nn` is a number between 1 and the value that is specified for the SORTNUM installation option or the OPTIONS SORTNUM parameter.

  When you use dynamic allocation for these files, RECOVER PLUS determines the optimal number of files to use.

  **NOTE**

  When the value of MAXKSORT is greater than 1, RECOVER PLUS ignores the WORKDDN option on the REBUILD INDEX command and issues a warning message. To use the WORKDDN option, specify a value of 1 for MAXKSORT. For a description of WORKDDN, see “WORKDDN DDName” on page 221.

**MAXLOGS=5**

This option specifies the default value for the maximum number of log files that RECOVER PLUS reads at one time. MAXLOGS must be greater than 0. The default value is 5. You can override the MAXLOGS installation option by using the MAXLOGS keyword with the OPTIONS command (see page 121).

**MAXDRIVE=0**

This option specifies the default value for the maximum number of tape drives to use during a utility run. The default value is 0, which means tape drive usage is unlimited. You can override the MAXDRIVE installation option by using the MAXDRIVES keyword with the OPTIONS command (see page 120).

**SMCORE=(0K,0K)**

This option specifies the amount of memory that you want each invocation of BMCSORT to use. For more information, see “Managing sort performance” on page 433.

- The first value specifies the total amount of memory used both below and above the 16-MB line.

- The second value specifies only the amount of memory used below the 16-MB line.

  You can enter either value as a number of kilobytes (using the suffix K) or megabytes (using the suffix M).
BMC recommends that you specify zero for both SMCORE values (as 0K or 0M) to allow RECOVER PLUS to determine the optimal amount of storage to allocate above and below the 16-MB line based on the following criteria:

- number of sorts to process
- amount of memory that is available during optimization
- value that you specified for REGION in either your JCL or system exits

For the first value, the following values are valid:

- zero, specified as 0K or 0M, tells RECOVER PLUS to determine the appropriate amount
- 4096 KB through 65536 KB (specified by using the K suffix) or 4 MB through 64 MB (specified by using the M suffix) tells RECOVER PLUS to use the specified amount

For the second value, the following values are valid:

- 0K (or 0M) tells RECOVER PLUS to determine the appropriate amount.
- 256 KB through 4096 KB (specified by using the K suffix) or 1 MB through 4 MB (specified by using the M suffix) tells RECOVER PLUS to use the specified amount

**RESINV=0K**

RESINV specifies the amount of memory reserved below the 16-MB line by the sort routine to allow for IDCAMS processing. The default value of RESINV=0K allows the sort routine defaults to be used. **BMC recommends that you use the default value so that the sort routine uses its algorithms effectively.**

**WKUNIT=SYSALLDA**

The unit for work data set dynamically allocated by RECOVER PLUS (for example, for DSNUTILB or BMCSORT). If you do not want to use the default value, SYSALLDA, you can change to some other appropriate generic name. Ensure that the unit specified is a valid one in your installation. Do not specify VIO for this option.

**XBMID=ssid or xbmGroup**

The XBMID option specifies the EXTENDED BUFFER MANAGER (XBM) subsystem ID (ssid) or XBM group name (xbmGroup) to use when you restore Instant Snapshot copies or when you use the zIIP redirection capability.

ssid is the unique identifier that you specified when you installed XBM. If you are using XBM in a DB2 data sharing environment, you can use the xbmGroup name in place of ssid. The xbmGroup value is the name of the XBM coupling facility group defined to the XBM subsystem.
For the zIIP redirection capability, if you specify an XBM subsystem and ZIIP ENABLED (page 130 and page 547) is in effect, RECOVER PLUS attempts to use that subsystem to enable zIIP processing. If that subsystem is not available or if it is not at the correct maintenance level, zIIP processing is not enabled.

If you do not specify an XBM subsystem either with XBMID on the OPTIONS command or with the XBMID installation option, RECOVER PLUS searches for an XBM subsystem at the appropriate maintenance level to enable zIIP processing.

For recovery using Instant Snapshots, RECOVER PLUS does not discover the XBM subsystem. Use the XBMID option in either the installation options module or on the OPTIONS statement to specify it. Instant Snapshots are made by COPY PLUS with XBM and are registered in BMCXCOPY. For more information, see the COPY PLUS for DB2 Reference Manual.

You can override the XBMID installation option by using the XBMID option with the OPTIONS command (see page 129).

For specific information about valid characters for XBMID and the pattern matching capabilities of XBM, Instant Snapshots, and zIIP redirection, see the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide.

**ZIIP=ENABLED**

The ZIIP option tells RECOVER PLUS whether to attempt to use IBM® System z® Integrated Information Processors (zIIPs). RECOVER PLUS can use enclave service request blocks (SRBs) to enable zIIP processing automatically while running jobs. Using zIIP processing can reduce the overall CPU time for RECOVER PLUS jobs.

You can specify one of the following values:

- **ENABLED** tells RECOVER PLUS to attempt to offload eligible processing to an available zIIP. If the zIIP is busy or not available, normal processing continues on a general-purpose processor.

- **DISABLED** tells RECOVER PLUS to not attempt to use zIIP processing.

To enable and use zIIP processing with RECOVER PLUS, you must

- have an installed authorized version of XBM or SUF
- start and maintain an XBM subsystem in your environment
- have a zIIP available in your environment

You can specify a particular XBM subsystem to use by specifying a value for the XBMID installation option or OPTIONS command XBMID option. For more information, see “XBMID=ssid or xbmGroup” on page 546 or page 129.
XBM and SUF are licensed, installed, and maintained separately from RECOVER PLUS. You can use either XBM or SUF, depending on the license that you have obtained:

- A license for the full version of the XBM product authorizes you to use all features of XBM.
- A license for SUF authorizes you to use only the snapshot and zIIP-processing features of XBM.

For more information about XBM and SUF, see the *EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide*.

You can override the value for this option by using the ZIIP command option (page 130).

**WTOR=YES**

If one or more spaces in the RECOVER command statement remain in STOPP status, WTOR=YES provides the flexibility to issue a write to the operator with reply (WTOR), while WTOR=NO assumes an operator reply of CANCEL and terminates the job.

**ERRCONT=10**

This option specifies the maximum number of times to continue processing after encountering a severe error. ERRCONT + 1 errors could occur before RECOVER PLUS terminates.

**TZRULE=NULL**

The TZRULE installation option defines the rule to determine when DST begins and ends, enabling RECOVER PLUS to calculate the dates and times for the changes each year. Using TZRULE, you do not need to update the value.

TZRULE enables RECOVER PLUS to adjust day and time values when you run the product after a DST change, but scan log records that are from a period before the time change, which is called scanning across a DST boundary.

The default value, NONE, indicates that your locale does not observe DST or that you do not want the product to make adjustments for DST. To enable DST adjustment, you must define day and time values as a character string in the following format:

TZRULE=startDayRule[/time],endDayRule[/time]
The first date (*startDayRule*) describes when the change to DST occurs, and the second date (*endDayRule*) describes when the change back happens. The time is optional and defaults to 02:00:00 (the default conversion time).

Use one of the following formats for *startDayRule* and *endDayRule*:

- **Mm.n.d**

  This is the standard format for most countries. Using this format, specify the month, week, and day of the week on which DST begins and ends, as follows:

  — The variable *m* represents the month; valid values are 1 through 12 (January through December).

  — The variable *n* represents the week of the month in which the day *d* occurs; valid values are 1 through 5. Week 1 is the first week in which day *d* occurs, and week 5 specifies the last *d* day in the month.

  — The variable *d* represents the day; valid values are 0 through 6 (Sunday through Saturday).

  In the following example, DST begins on the second Sunday of March at 2:00 a.m., and ends on the first Sunday of November at 2:00 a.m.:

  ```
  TZRULE=M3.2.0/2:00:00,M11.1.0/2:00:00
  ```

  The following example, which uses the default conversion time of 2:00 a.m and omits the time specification, is also valid:

  ```
  TZRULE=M3.2.0,M11.1.0
  ```

- **Jn**

  Specify the Julian day *n*, where *n* is a value from 1 through 365. This format does not account for leap days; that is, in all years, including leap years, February 28 is day 59 and March 1 is day 60. You cannot explicitly refer to February 29.

  In the following example, DST begins on April 1 at 12:34:56 p.m., and ends on November 1 at 2:34:56 a.m.:

  ```
  TZRULE=J91/12:34:56,J305/2:34:56
  ```
Considerations for DST adjustment

To understand DST adjustment, remember that RECOVER PLUS uses the operating system to translate date and time values into the operating system’s store clock format. The operating system uses an offset based on your time zone to translate the values. For example: assume that you are in the northern hemisphere, you run RECOVER PLUS in May (after the change to DST), and you define a scan range from 15:00 to 16:00 on February 14 (before the time change).

If you set TZRULE to enable adjustment, the product knows that the time zone offset in May is different than the time zone offset that was used when DB2 wrote the log records in February (the difference is one hour). The product adjusts for this difference, selects log records from 15:00 to 16:00 on February 14, and prints the time values of those records in reports starting with 15:mm.

If you do not set TZRULE to enable adjustment, the product uses the time zone offset for May. Because the offset was different in February, the product selects log records from 14:00 to 15:00 on February 14, and prints the time value of those log records in reports starting with 15:mm. The discrepancy in the time zone offset creates a discrepancy between the log records that you request and the log records that the product provides in output.

Be aware of the following additional points regarding DST adjustment and the TZRULE installation option:

- RECOVER PLUS provides adjustment because the operating system does not account for changes in the time zone offset that have occurred in the past. The operating system always uses the current time zone offset when it converts a date and time value into store clock format.

- By default, adjustment is not enabled. To enable it, set the TZRULE installation option value.

- The options migration feature of the Installation System cannot carry over DST installation option values from a previous release of the product. You must provide a value for TZRULE if you want to enable DST adjustment.

- Adjustment affects only how the product handles the standard timestamp values that DB2 stores in all log records (using the operating system’s store clock format).
LOCCPSEL=(FC,LP,LB)

This option specifies the sequence of image copy files for selection for local site recovery. The default sequence is IBM FlashCopy (FC), local primary (LP), then local secondary (LB). You can specify two or three values from FC, LP, LB, remote primary (RP), and remote secondary (RB) in any order.

LOCCASEL=(LP,LB)

This option specifies the sequence of change accumulation files for selection for local site recovery. The default sequence is local primary (LP), and then local secondary (LB). You can specify a pair of values from LP, LB, remote primary (RP), and remote secondary (RB) in any order.

REMCPSEL=(RP,RB,FC)

This option specifies the sequence of image copy files for selection for remote site recovery. The default sequence is remote primary (RP), remote secondary (RB), and then IBM FlashCopy (FC). You can specify two or three values from RP, RB, FC, local primary (LP), and local secondary (LB) in any order.

REMCASEL=(RP,RB)

This option specifies the sequence of change accumulation files for selection for remote site recovery. The default sequence is remote primary (RP), then remote secondary (RB). You can specify a pair of values from RP, RB, local primary (LP), and local secondary (LB) in any order.

AUTOSIZE=YES

This option turns dynamic sizing for change accumulation output files on or off. Valid values are YES, Y, NO, and N. The default value is YES. AUTOSIZE=YES means dynamic sizing of change accumulation output files to DASD occurs. AUTOSIZE=NO means change accumulation output files to DASD are allocated using the primary and secondary quantities that are specified in the R+/CHANGE ACCUM repository.

You can override the AUTOSIZE installation option by using the AUTOSIZE option on the OPTIONS command (see page 140).
**KEYDSNAM= keyDataSetName**

The KEYDSNAM option specifies the name of the key data set that is used for encrypted copies.

For more information about encrypted copies and the key data set, see “Recovering encrypted copies” on page 462.

---

**NOTE**

Encryption is a feature of the Recovery Management for DB2 solution and requires a valid Recovery Management solution password.

---

**ACFORTSS=YES**

The ACFORTSS option specifies whether RECOVER PLUS should look and use CA ACF2 security or CA Top Secret security from Computer Associates when verifying utility authorizations. Valid values are NO, N, YES and Y.

If you specify YES, RECOVER PLUS looks for and uses CA ACF2 security or CA Top Secret security provided that security is enabled at the correct version on the subsystem where RECOVER PLUS is running. This is the default value. If RECOVER PLUS cannot find one of these security products, it uses DB2 native security.

You must be using a version of your security product that enables external security calls for DB2. If you have one of these security products installed, but the version does not support external security, specify NO for this option.

If you specify NO, RECOVER PLUS does not look for or use CA ACF2 security or CA Top Secret security, but uses DB2 native security.

For more information, see “System authority” on page 81.

---

**NOTE**

If you do not have CA ACF2 security or CA Top Secret security, this option has no effect.

---

**DATAMVR=**

The DATAMVR installation option provides XBM with the name of the program to use to copy a data set if a data set snap fails. To use DFDSS as the data mover, specify DATAMVR=ADRDSSU. You can override the DATAMVR installation option by using the DATAMVR parameter with the OPTIONS command (see page 131).
**BINDQUALIFIER=AFR<vv>r**

Use the BINDQUALIFIER installation option to specify the RECOVER PLUS bind qualifier for the dynamic bind process. The bind qualifier determines which set of synonyms RECOVER PLUS is to use. The BINDQUALIFIER value can have a maximum length 16 bytes. The default value is AFR<vv>r, where <vv> represents a two-character version number and <r> represents a one-character release number, such as AFR101.

**PUBLICPLAN=YES**

Use the PUBLICPLAN installation option to grant the PUBLIC privilege to run RECOVER PLUS. Valid values for PUBLICPLAN are YES and NO. The default value is YES. If you set the value of this option to NO, the installation process will not do any grants, which means that you must grant execute authority to users as needed.

**HISTORY=YES**

The HISTORY option specifies whether RECOVER PLUS is to update the BMCHIST table. Valid values are YES and NO. The default value is YES. If you specify YES, RECOVER PLUS updates the BMCHIST table by writing one row per RECOVER PLUS job step. If you specify NO, RECOVER PLUS bypasses any updates to the BMCHIST table.

See Appendix C, “BMC utilities database,” for more information about the BMCHIST table.

**MAXLSORT=0**

This option specifies the maximum number of log sorts that can run concurrently and also determines the number of MERGE/RESTORE/SNAP phases that can run in parallel, whether or not log records are processed. You use MAXLSORT to improve recovery performance.

Valid values are 0 to 999. When MAXLSORT=0, the default value is determined by RECOVER PLUS.

---

**NOTE**

BMC suggests values from 10 to 12 for MAXLSORT.

You can override the MAXLSORT installation option by using the MAXLSORT parameter on the OPTIONS command (see page 136).
When you specify MAXLSORT, the following files are dynamically allocated if you do not code them in JCL:

- **LOGOU**\textit{nnn}: sort message files
  
  \textit{nnn} is the number of the log sort and is a number between 1 and the value that is specified for MAXLSORT.

- **L**\textit{xxxWK**\textit{nnn}: sort work files
  
  \textit{xxx} is the number of the log sort and is a number between 1 and the value that is specified for MAXLSORT. \textit{nn} is the number of the work data set. For example, if MAXLSORT=3 and two sort work files are required for each sort, the DDs would be specified as follows:

<table>
<thead>
<tr>
<th>DD Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>L001WK01 DD...</td>
</tr>
<tr>
<td>L001WK02 DD...</td>
</tr>
<tr>
<td>L002WK01 DD...</td>
</tr>
<tr>
<td>L002WK02 DD...</td>
</tr>
<tr>
<td>L003WK01 DD...</td>
</tr>
<tr>
<td>L003WK02 DD...</td>
</tr>
</tbody>
</table>

When you use dynamic allocation for these files, RECOVER PLUS determines the optimal number of files to use.

**KSORTSHARE=\textbf{YES}**

This option specifies if key sorts are shared among RECOVER PLUS table space recoveries (MERGE phases) running in parallel.

When you specify YES, which is the default value, RECOVER PLUS uses up to the value specified for MAXKSORT active key sorts at any given time. If sufficient key sorts are not available when a table space recovery begins execution, keys will be obtained later by an UNLOAD phase.

When you specify NO, each MERGE phase has its own set of key sorts and up to MAXKSORT * MAXLSORT key sorts can be active at any given time. Since the number of sorts that can be active in a system is fairly small – usually no more than 30 – a value of NO for this option may severely limit the number of recovery operations that RECOVER PLUS can perform in parallel when index rebuilds are also requested.

You can override the KSORTSHARE installation option by using the KSORTSHARE parameter on the OPTIONS command (see page 136).
AUX=NO

Use the AUX installation option to specify if the auxiliary objects related to a base table space are to be included in recovery. Valid values are NO, YES, XML, LOB, and HISTORY. The default value is NO. These values have the following meanings:

- **AUX=NO**: Do not include any auxiliary objects in the recovery.
- **AUX=YES**: Includes both LOB and XML objects in the recovery, as well as history spaces when a table space with system-maintained temporal tables is specified in SYSIN.
- **AUX=XML**: Includes all XML data objects along with the XML base table space. If the table space specification is found on a RECOVER TABLESPACE or SIMULATE TABLESPACE statement, all implicitly created XML table spaces will be processed along with the base XML table space. If the table space specification is found on a RECOVER INDEX(ALL), REBUILD INDEX(ALL), or SIMULATE INDEX(ALL) statement, all implicitly created XML and node ID indexes on the XML base table space as well as all explicitly created non-LOB indexes will be processed.
- **AUX=LOB**: Includes all LOB data objects along with the LOB base table space. If the table space specification is found on a RECOVER TABLESPACE or SIMULATE TABLESPACE statement, all table spaces created with the LOB attribute will be processed along with the base LOB table space. If the table space specification is found on a RECOVER INDEX(ALL), REBUILD INDEX(ALL) or SIMULATE INDEX(ALL) statement, all indexes on the LOB base table space and all non-LOB explicitly created indexes will be processed.
- **AUX=HISTORY**: Includes history spaces when a table space with system-maintained temporal tables is specified in SYSIN. Similar to DB2 RECOVER Version 10 VERIFYSET NO behavior, RECOVER PLUS does not check for the following:
  - that the system-maintained temporal space and the history space are recovered as a set
  - that the recovery of the system-maintained temporal space and the history space is to the same log point
  - that all DSNUMs are recovered in the same SYSIN
You can override the value of the AUX installation option at runtime by using the AUX option on the OPTIONS command (page 122) or in the table space specification for any of the following RECOVER PLUS statements:

- RECOVER TABLESPACE
- RECOVER INDEX(ALL) TABLESPACE
- REBUILD INDEX(ALL) TABLESPACE
- LOGSCAN TABLESPACE
- LOGSCAN INDEX(ALL) TABLESPACE

**NOTE**

RECOVER PLUS ignores the AUX option when you specify OBJECTSET.

Also, the AUX option has no effect for the ACCUM TABLESPACE, RECOVER UNLOADKEYS, and RECOVER BUILDINDEX commands.

When you specify AUX=YES, AUX=XML, AUX=LOB, or AUX=HISTORY, you cannot specify the following options:

- TOCOPY dataSetName
- OBIDXLAT without defaults
- OUTCOPYDSN
- RECOVERYDSN

**BACKOUT=AUTO**

This option specifies the default for BACKOUT recovery. Valid values are AUTO and NO. AUTO is the default value. With this default value, if you are running with a Recovery Management password, any point-in-time recovery uses BACKOUT AUTO. If you are not using a Recovery Management password, point-in-time recoveries use a standard forward recovery.

You can override the value of the BACKOUT installation option at runtime by using the BACKOUT option on the OPTIONS command (page 124).

For more information about BACKOUT AUTO recovery, see “Using BACKOUT AUTO recovery” on page 61.

If you do not want BACKOUT AUTO to be the default value, change the BACKOUT installation option to BACKOUT NO.

If the site type in DSNZPARM is a recovery site, RECOVER PLUS uses BACKOUT NO as the default value. If you are doing disaster recovery at local site (site type is local site in DSNZPARM), you should set the BACKOUT installation option to NO for disaster recovery.
USEHDROBIDS=YES

Use USEHDROBIDS to indicate if the OBIDs in the header are valid or not. Valid values are YES, the default, and NO.

When you specify OBIDXLAT (page 194) with INCOPY FULL SNAPSHOT TOCOPY syntax and do not specify OBIDs, RECOVER PLUS looks at the OBIDs in the header page to determine if the OBIDs for the source and target are the same. If the OBIDs are the same, RECOVER PLUS does not use a MERGE phase to update the OBIDs. Skipping the MERGE phase makes migration faster because it eliminates a read and write of every page of the space. (RECOVER PLUS may still do an "optimized merge" phase to just update the header page.)

Sometimes the OBIDs in the header page are not correct. An example of when this situation might occur is when the space is copied from another object and the OBIDs in the header page are not translated correctly. If the OBIDs in the header page are not correct, RECOVER PLUS might skip the MERGE phase when it should not, or the MERGE phase might fail because RECOVER PLUS detects that the OBIDs in the header page do not match the OBIDs in the rest of the data. Under circumstances like these, you can use USEHDROBIDS to tell RECOVER PLUS if the OBIDs in the header are valid or not.

You can override the USEHDROBIDS installation option by using the USEHDROBIDS parameter on the OPTIONS command (see page 123).

USEHDROBIDS YES

When you specify USEHDROBIDS YES, RECOVER PLUS uses the OBIDs in the header page and skips the MERGE phase if possible. If RECOVER PLUS does perform the MERGE phase and detects that the OBIDs are not correct, RECOVER PLUS ends with an error message.

USEHDROBIDS NO

When you specify USEHDROBIDS NO, RECOVER PLUS always does a MERGE phase and ignores the OBIDs in the header page.

EATTR=

Use EATTR to specify whether a data set supports extended attributes or not. If EATTR is not specified, which is the default, an SMS DATACLAS can provide the value.

You cannot use an image copy made to the cylinder-managed portion of an extended address volume (EAV) under z/OS Version 1.11 on z/OS Version 1.10 because z/OS Version 1.10 does not support sequential data sets in the cylinder-managed portion of an EAV.
z/OS Version 1.11 or later supports the EATTR option. For earlier versions of z/OS, you must set EATTR= (or EATTR=NONE).

If you are using DB2 Version 9 and z/OS Version 1.11, RECOVER PLUS can create a data set with extended attributes and will register it in SYSCOPY. However, the DB2 RECOVER Version 10 utility is required to work with the registered data set.

You can set EATTR to OPT or NO in the JCL.

When you create an OUTCOPY (page 199) and want to force the copy to the cylinder-managed space of an EAV, specify the SPACE keyword (page 204).

Valid values for EATTR are:

- Specifying no value for EATTR (EATTR=), the default, allows the value for EATTR to be set by an SMS DATACLAS. (EATTR= is the same as specifying EATTR=NONE.)

  Using the default value allows you to have your environment set up to use extended attributes.

- OPT specifies that extended attributes are optional for the data set.

  You must set EATTR=OPT to allocate an extended format sequential data set. By using EATTR=OPT, RECOVER PLUS supports sequential data sets in the cylinder-managed portion of EAVs.

  If you specify EATTR=OPT, RECOVER PLUS specifies the EATTR attribute when it dynamically allocates the output data set and overrides the EATTR option in the SMS DATACLAS, if one exists.

  Extended format sequential data sets must be allocated on SMS-managed volumes and the size of the data set must be greater than the EAV break point, which is typically 10 cylinders.

- NO specifies that the data set cannot have extended attributes.

  If you specify EATTR=NO, RECOVER PLUS specifies the EATTR attribute when it dynamically allocates the output data set and overrides the EATTR option in the SMS DATACLAS, if one exists.

  By using the EATTR syntax option in an OUTPUT statement at runtime, you can override the value set during installation. (See page 151.)
**DATACLAS=**

Specify DATACLAS= when you want to provide an SMS data class name. The value of name must be a valid SMS data class name, not exceeding eight characters.

RECOVER PLUS forces CATLG=YES when you specify DATACLAS.

By using the DATACLAS syntax option in an OUTPUT statement at runtime, you can override the value set during installation. (See page 150.)

**STORCLAS=**

Specify STORCLAS= when you want to provide an SMS storage class name. The value of name must be a valid SMS storage class name, not exceeding eight characters.

RECOVER PLUS forces CATLG=YES when you specify STORCLAS.

By using the STORCLAS syntax option in an OUTPUT statement at runtime, you can override the value set during installation. (See page 150.)
BMCSORT installation and options

This appendix presents the following topics:

Installation overview ................................................................. 561
DYNALOC Installation Option ...................................................... 561

Installation overview

BMCSORT is installed automatically when you install a product or solution that invokes BMCSORT. You do not need a password to run BMCSORT.

BMCSORT requires that you specify a value for the installation option DYNALOC. For more information about the DYNALOC option, see “DYNALOC Installation Option.”

DYNALOC Installation Option

The DYNALOC installation option provides information for dynamically allocating SORTWK data sets. BMCSORT deallocates these data sets at the end of each sort. The content of the $AUPSMAC macro in $C32SOPT follows, showing DYNALOC and the values that are shipped with BMCSORT.

```
$AUPSMAC DYNALOC=(SYSDA,3,ON,ON,6000000,3000000,3390,SC=,RETRY=(0,0)) X
```

The values that you specify in this macro apply to all invocations of BMCSORT. BMCSORT uses the same options module for all BMC Software products that invoke BMCSORT. You can have only one options module for BMCSORT.
Table 20 on page 563 describes each parameter of the DYNALOC option. These parameters are positional. The values that you specify for these parameters should correspond to your site’s standards for any system sort routine.

BMCSORT overrides the values that you supplied if BMCSORT determines that it can complete sorting more efficiently than the specified values allow.

An invoking product’s options might override the BMCSORT options values that you specify as follows:

- the values in the invoking product’s dynamic allocation installation options or corresponding command options conflict with the values you specify

- you turn on BMCSORT SORTWK dynamic allocation using your product’s command and you specify OFF for the position 3 parameter.

  BMCSORT dynamically allocates SORTWK files as necessary.
<table>
<thead>
<tr>
<th>Parameter Name or Position</th>
<th>Description</th>
<th>Initial Value</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>position 1</td>
<td>This parameter is the generic unit name from which RECOVER PLUS should dynamically allocate SORTWK data sets. This parameter applies only when the Data Facility Storage Management System (DFSMS™) product from IBM is not installed or is not active for temporary DASD work data sets. If DFSMS is active, use the SC parameter.</td>
<td>SYSDA</td>
<td>Use a unit name up to 8 characters.</td>
</tr>
<tr>
<td>position 2</td>
<td>Do not change this value. RECOVER PLUS does not use this parameter, but it must exist for proper assembly of the installation options macro.</td>
<td>3</td>
<td>Do not change this value. Changing this value can prevent proper assembly of the installation options macro.</td>
</tr>
<tr>
<td>position 3</td>
<td>This parameter tells RECOVER PLUS whether to dynamically allocate SORTWK files. <strong>Note:</strong> BMC Software recommends that you not change this value.</td>
<td>ON</td>
<td></td>
</tr>
<tr>
<td>position 4</td>
<td>Do not change this value. RECOVER PLUS does not use this parameter, but it must exist for proper assembly of the installation options macro.</td>
<td>ON</td>
<td>Do not change this value. Changing this value can prevent proper assembly of the installation options macro.</td>
</tr>
<tr>
<td>position 5</td>
<td>Do not change this value. RECOVER PLUS does not use this parameter, but it must exist for proper assembly of the installation options macro.</td>
<td>6000000</td>
<td>Do not change this value. Changing this value can prevent proper assembly of the installation options macro.</td>
</tr>
<tr>
<td>position 6</td>
<td>Do not change this value. RECOVER PLUS does not use this parameter, but it must exist for proper assembly of the installation options macro.</td>
<td>3000000</td>
<td>Do not change this value. Changing this value can prevent proper assembly of the installation options macro.</td>
</tr>
<tr>
<td>position 7</td>
<td>This parameter specifies the DASD type with the smallest track capacity that a dynamically allocated SORTWK data set might encounter at your site.</td>
<td>3390</td>
<td>3380, track capacity of 47968 3390, track capacity of 56664 9345, track capacity of 46456</td>
</tr>
</tbody>
</table>
### Table 20  DYNALOC parameters (part 2 of 2)

<table>
<thead>
<tr>
<th>Parameter Name or Position</th>
<th>Description</th>
<th>Initial Value</th>
<th>Valid Values</th>
</tr>
</thead>
</table>
| SC | This parameter specifies the name of the DFSMS storage class from which to dynamically allocate SORTWK. If DFSMS is active and you do not specify a value for this parameter, RECOVER PLUS uses the value from the first DYNALOC parameter.  

**Note:** If your installation has an automatic class selection (ACS) routine, it can override this specification. | blank | Use any valid DFSMS storage class. |
| RETRY | This parameter specifies how you want RECOVER PLUS to handle retry attempts for SORTWK dynamic allocation. The first subparameter indicates the number of times that you want RECOVER PLUS to retry the request. The second subparameter indicates the number of minutes to wait between each retry.  

Using this parameter allows you to avoid a capacity exceeded condition when disk space is not immediately available for a SORTWK dynamic allocation request.  

BMC Software recommends that you do not change this value because it can affect the elapsed time of your jobs. However, if you currently use SyncSort and you rely on this function, BMC Software recommends that you use the same values as your SyncSort RETRY installation parameter. | (0,0) | If you use this parameter, BMC Software recommends that you specify the same values as your SyncSort RETRY installation parameter. The following values are valid for this parameter:  

- first subparameter—0 through 16  
  A value of 0 indicates that you do not want RECOVER PLUS to retry the request.  
- second subparameter—0 through 15  
  A value of 0 indicates that you do not want RECOVER PLUS to retry the request. |
BMC utilities database

This appendix presents the following topics:

Overview ................................................................. 566
  Considerations and warnings ........................................ 566
  Managing common utility tables .................................. 567
BMCDICT table ......................................................... 569
  Considerations ....................................................... 569
  Maintaining the BMCDICT table ................................... 570
BMCHIST table ......................................................... 570
  Maintaining the BMCHIST table ................................... 572
BMCLGRNX table ....................................................... 572
BMCSYNC table ......................................................... 573
  Executing BMC utilities concurrently .............................. 575
  Considerations ....................................................... 577
  Maintaining the BMCSYNC table ................................... 577
  Cleaning up RECOVER UNLOADKEYS entries ...................... 578
BMCTRANS table ....................................................... 578
BMCUTIL table ......................................................... 580
  Maintaining the BMCUTIL table ................................... 582
BMXCOPY table ......................................................... 582
  Maintaining the BMXCOPY table ................................... 586
Overview

The BMC common utility tables contain information about the BMC utilities that you generate and submit through a BMC utility product. Table 21 lists the tables that each utility uses and each table's default name and synonym.

Table 21 Common utility tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Default name</th>
<th>Synonym</th>
<th>CHECK PLUS</th>
<th>COPY PLUS</th>
<th>DASD MANAGER PLUS (BMCSTATS)</th>
<th>LOADPLUS</th>
<th>RECOVER PLUS</th>
<th>RECOVERY MANAGER</th>
<th>REORG PLUS</th>
<th>UNLOAD PLUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMCDICT</td>
<td>CMN_BMCDICT</td>
<td>BMC_BMCDICT</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMCHIST</td>
<td>CMN_BMCHIST</td>
<td>BMC_HISTORY</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMCLGRNX</td>
<td>CMN_BMCLGRNX</td>
<td>BMC_BMCLGRNX</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMCSYNC</td>
<td>CMN_BMCSYNC</td>
<td>BMC_BMCSYNC</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMCTRANS</td>
<td>CMN_BMCTRANS</td>
<td>BMC_BMCTRANS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMCUTIL</td>
<td>CMN_BMCUTIL</td>
<td>BMC_BMCUTIL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMCXCOPY</td>
<td>CMN_BMCXCOPY</td>
<td>BMC_IXCOPY</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Considerations and warnings

Note the following considerations when using the common utility tables:

- Some columns in the tables are present for compatibility with specific BMC utilities and are not used by all of the utilities.

- If you have applications that depend on the structure or content of these tables, be aware that these tables are subject to change.

- In general, the utility tables should not require maintenance, with the exception of BMCHIST.
You should back up the BMC table spaces on a regular basis to enable recoveries. If you use COPY PLUS as the copy utility, you must use SHRLEVEL CHANGE for the following spaces:

- BMCUTIL
- BMCHIST
- BMCSYNC
- BMXCOPY

**WARNING**

The following warnings apply:

- Do not run LOADPLUS, REORG PLUS, or UNLOAD PLUS against the BMC common utility tables or table spaces. Doing so can cause unpredictable results.

- Because RECOVER PLUS uses BMC tables during the recovery process, you cannot use RECOVER PLUS to recover the BMC tables, with the exception of BMCHIST.

- Do not run the RUNSTATS utility against the BMC common utility tables. Doing so can negatively impact utility performance.

- BMC strongly recommends that you use the ISOLATION (UR) bind option and issue SQL COMMIT statements when querying the tables in the BMC database. If objects in the BMC database are restricted for UPDATE, the executing BMC utilities might not be able to complete successfully.

### Managing common utility tables

This section provides basic procedures for working with the common utility tables.

**To determine your site’s table names**

The names of the common utility tables can be changed during installation. To determine the names that your site uses, perform one of the following actions:

- Use your utility to run a job with restart parameters of MAINT and MSGLEVEL(1).

  Specifying MSGLEVEL(1) with MAINT prints the names of the BMC tables that your utility uses and identifies the applied maintenance. The utility does not perform any other processing, and the job ends without affecting any utility that is running.
Managing common utility tables

- Get the names from your DB2 system administrator.

- Run the following SQL statement, replacing `tableName` with a BMC common utility table name (for example, CMN_BMCHIST or CMN_BMSYNC):

```
SELECT CREATOR, NAME FROM SYSTBL.SYSTABLES
WHERE TSNAME = 'tableName';
```

**To query the tables**

Run SQL statements similar to the following examples.

---

**EXAMPLE**

This example queries the BMCXCOPY table to access information about the rows in an index space:

```
SELECT *
FROM creatorName.CMN_BMCXCOPY
WHERE DBNAME = 'databaseName'
  AND IXNAME = 'indexSpaceName'
ORDER BY START_RBA;
```

This example identifies (from the BMCHIST table) the database name, table space name, elapsed time, and when the utility completed:

```
SELECT DBNAME, SPNAME, CHAR(ELAPSED,ISO), CHAR(TIME,ISO)
FROM creatorName.CMN_BMCHIST
WHERE UTILID = 'utilityID';
```

---

**To display BMC utility status**

To display the status of all BMC utilities that are executing or awaiting restart for a given table space or index space, use the following SQL statements:

```
SELECT * FROM creatorName.CMN_BMCMUTIL
WHERE DBNAME = 'databaseName'
  AND SPNAME = 'tableSpaceName'
SELECT * FROM creatorName.CMN_BMCSYNC
WHERE NAME1 = 'databaseName'
  AND NAME2 = 'spaceName';
```
To terminate a BMC utility

To terminate a BMC utility that is executing, use the following SQL statements:

```
DELETE FROM creatorName.CMN_BMCUTIL
WHERE UTILID='utilityID';
DELETE FROM creatorName.CMN_BMCSYNC
WHERE UTILID='utilityID';
DELETE FROM creatorName.CMN_BMCDICT  -- for LOADPLUS and REORG PLUS
WHERE UTILID='utilityID';
```

The utility terminates with return code 8 when the next checkpoint is taken.

To clean up a BMC utility that is not executing, run the utility with the correct utility ID and specify TERM as the restart parameter.

BMCDICT table

Table 22 describes the BMCDICT table, which stores the compression dictionary during load or reorganization processing.

### Table 22  BMCDICT table

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTILID</td>
<td>CHAR(16)</td>
<td>utility identifier</td>
</tr>
<tr>
<td>DBNAME</td>
<td>CHAR(8)</td>
<td>database name</td>
</tr>
<tr>
<td>TSNAME</td>
<td>CHAR(8)</td>
<td>table space name</td>
</tr>
<tr>
<td>PARTITION</td>
<td>SMALLINT</td>
<td>partition number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For a nonpartitioned table space, the value is 0.</td>
</tr>
<tr>
<td>SEQNO</td>
<td>SMALLINT</td>
<td>sequence number</td>
</tr>
<tr>
<td>DICTDATA</td>
<td>VARCHAR(4000)</td>
<td>dictionary data</td>
</tr>
</tbody>
</table>

Considerations

Note the following considerations:

- If you are processing a large number of compressed partitions, you might need to increase the size of the BMCDICT table space significantly from the standard size that was allocated during installation. To estimate the allocation, multiply 64 KB by the number of compressed partitions that you are processing concurrently (loading with LOADPLUS and reorganizing with REORG PLUS).
Maintaining the BMCDICT table

- LOADPLUS inserts rows into the BMCDICT table during the PRELOAD phase and deletes those rows following compression processing in the LOAD phase.

- REORG PLUS inserts rows into the BMCDICT table during the UNLOAD phase and deletes those rows following compression processing in the RELOAD phase.

Maintaining the BMCDICT table

If LOADPLUS or REORG PLUS abends during the time between building the compression dictionary and completing compression, rows might remain in the BMCDICT table. If you need to control the expansion of this table, use the following procedure:

1. Delete any rows in the BMCUTIL table that you know are no longer valid.
   
   Do not delete any rows for instances of utilities that are awaiting restart.

2. Use the following SQL statement to delete rows from the BMCDICT table:

   ```sql
   DELETE FROM creatorName.CMN_BMCDICT
   WHERE UTILID NOT IN
   (SELECT UTILID FROM creatorName.CMN_BMCUTIL);
   ```

   **NOTE**

   The names of the BMCUTIL and BMCDICT tables might have been changed at your site during installation.

BMCHIST table

Table 23 on page 571 describes the BMCHIST table, which contains information about completed executions of the BMC utilities for DB2. The following installation options control use of the BMCHIST table:

- HISTORY (for COPY PLUS, RECOVER PLUS, and UNLOAD PLUS)
- BMCHIST (for REORG PLUS)

If the option value is NO, the utility bypasses any updates to the BMCHIST table. If the value is YES (or the utility does not use an installation option), the utility inserts rows into the BMCHIST table during the UTILTERM phase.
Table 23  BMCHIST table

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBNAME</td>
<td>CHAR(8)</td>
<td>name of the database that contains the table or index space</td>
</tr>
<tr>
<td>SPNAME</td>
<td>CHAR(8)</td>
<td>name of the table or index space</td>
</tr>
<tr>
<td>UTILNAME</td>
<td>CHAR(8)</td>
<td>name of the utility:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- CHECK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- COPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- LOAD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- REORG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- UNLOAD</td>
</tr>
<tr>
<td>UTILID</td>
<td>CHAR(16)</td>
<td>utility identifier</td>
</tr>
<tr>
<td>AUTHID</td>
<td>CHAR(8)</td>
<td>user ID that ran the utility</td>
</tr>
<tr>
<td>DATE</td>
<td>DATE</td>
<td>date that the utility completed</td>
</tr>
<tr>
<td>TIME</td>
<td>TIME</td>
<td>time that the utility completed</td>
</tr>
<tr>
<td>ELAPSED</td>
<td>TIME</td>
<td>elapsed time of the utility</td>
</tr>
<tr>
<td>PARTITION</td>
<td>LONG VARCHAR</td>
<td>ALL, or the partition numbers as specified by the DSNUM option (for COPY PLUS) or the PART option</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note the following conditions:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- This column lists only three-digit partitions (any loaded partitions 1 through 999). Four-digit partitions (any loaded partitions from 1000 through 4096) are not stored in this column. For jobs that load only four-digit partitions, this column is empty.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- If the list of partitions exceeds 1011 bytes, the utility truncates the value that is stored in this column.</td>
</tr>
<tr>
<td>OBJNAME</td>
<td>VARCHAR(27)</td>
<td>fully qualified object name</td>
</tr>
<tr>
<td>PHASE_1</td>
<td>CHAR(8)</td>
<td>name of utility phase 1</td>
</tr>
<tr>
<td>ELAPSED_1</td>
<td>TIME</td>
<td>elapsed time of phase 1</td>
</tr>
<tr>
<td>PHASE_2</td>
<td>CHAR(8)</td>
<td>name of utility phase 2</td>
</tr>
<tr>
<td>ELAPSED_2</td>
<td>TIME</td>
<td>elapsed time of phase 2</td>
</tr>
<tr>
<td>PHASE_3</td>
<td>CHAR(8)</td>
<td>name of utility phase 3</td>
</tr>
<tr>
<td>ELAPSED_3</td>
<td>TIME</td>
<td>elapsed time of phase 3</td>
</tr>
<tr>
<td>PHASE_4</td>
<td>CHAR(8)</td>
<td>name of utility phase 4</td>
</tr>
<tr>
<td>ELAPSED_4</td>
<td>TIME</td>
<td>elapsed time of phase 4</td>
</tr>
<tr>
<td>PHASE_5</td>
<td>CHAR(8)</td>
<td>name of utility phase 5</td>
</tr>
<tr>
<td>ELAPSED_5</td>
<td>TIME</td>
<td>elapsed time of phase 5</td>
</tr>
</tbody>
</table>
Maintaining the BMCHIST table

When a utility completes successfully, it inserts a row into the BMCHIST table. Periodically, review BMCHIST and delete old rows to control its expansion.

To delete selected rows from the BMCHIST table based on the date that the utility completed, use the following sample SQL statement:

```
DELETE
FROM creatorName.CMN_BMCHIST
WHERE DATE < 'yyyy-mm-dd';
```

BMCLGRNX table

Table 24 describes the contents of the BMCLGRNX table, which contains log ranges that show when a table space was open for updates.

Table 24  BMCLGRNX table

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LGRDBID</td>
<td>CHAR(2)</td>
<td>DBID of the modified object</td>
</tr>
<tr>
<td>LGRPSID</td>
<td>CHAR(2)</td>
<td>OBID of the modified object</td>
</tr>
<tr>
<td>LGRUCDT</td>
<td>CHAR(6)</td>
<td>modification date (mmddyy)</td>
</tr>
<tr>
<td>LGRUCTM</td>
<td>CHAR(8)</td>
<td>modification time (hhmmssth)</td>
</tr>
<tr>
<td>LGRSRBA</td>
<td>CHAR(6)</td>
<td>starting RBA</td>
</tr>
<tr>
<td>LGRSPBA</td>
<td>CHAR(6)</td>
<td>stopping RBA</td>
</tr>
<tr>
<td>LGRPART</td>
<td>SMALLINT</td>
<td>table space partition number</td>
</tr>
<tr>
<td>LGRSLRSN</td>
<td>CHAR(6)</td>
<td>starting LRSN of update log records for data sharing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For non-data-sharing, the value is X'000000000000'.</td>
</tr>
<tr>
<td>LGRELRSN</td>
<td>CHAR(6)</td>
<td>ending LRSN of update log records for data sharing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For non-data-sharing, the value is X'000000000000'.</td>
</tr>
<tr>
<td>LGRMEMBER</td>
<td>CHAR(2)</td>
<td>data sharing member ID of the modifying DB2 subsystem</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For non-data-sharing, the value is X'0001'.</td>
</tr>
</tbody>
</table>
BMCSYNC table

Table 25 describes the BMCSYNC table, which contains information about the status of the objects that the currently executing utilities are accessing. The BMCSYNC table synchronizes and controls access to DB2 spaces by concurrently executing BMC utility products. If you have more than one BMC utility installed, all of these utilities should share the same BMCSYNC table.

The utilities insert rows into the BMCSYNC table during the UTILINIT phase. While the job executes, the utilities update the table as the status of the object changes. The utilities delete rows from the BMCSYNC table during the UTILTERM phase.

Table 25  BMCSYNC table (part 1 of 3)

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTILID</td>
<td>CHAR(16)</td>
<td>utility identifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For RECOVER PLUS, this column is blank when a RECOVER UNLOADKEYS command creates the row and then a RECOVER BUILDINDEX command reads and deletes the row.</td>
</tr>
<tr>
<td>NAME1</td>
<td>CHAR(8)</td>
<td>database name or creator name&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For DASD MANAGER PLUS, the value is the database name.</td>
</tr>
<tr>
<td>NAME2</td>
<td>CHAR(18)</td>
<td>space, table, or index name&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For DASD MANAGER PLUS, the BMCSTATS utility always inserts the space name (limited to a maximum of 8 characters).</td>
</tr>
<tr>
<td>KIND</td>
<td>CHAR(2)</td>
<td>type of object:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ IP (index partition)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ IX (index)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ TB (table)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ TP (table space partition)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ TS (table space)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ DD, DW (dynamic work file allocation)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ CI (copy information)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ RD (restart data set block)</td>
</tr>
<tr>
<td>PARTITION</td>
<td>SMALLINT</td>
<td>partition number:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ null or 0 for a single data set nonpartitioned space</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ data set number for a multi-data-set, nonpartitioned space</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ partition number for a partitioned space</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COPY PLUS, LOADPLUS, UNLOAD PLUS, CHECK PLUS, DASD MANAGER PLUS, and REORG PLUS use null or 0 for any nonpartitioned space.</td>
</tr>
<tr>
<td>BMCID</td>
<td>SMALLINT</td>
<td>internal identifier of the object</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DASD MANAGER PLUS does not use this column.</td>
</tr>
</tbody>
</table>

<sup>a</sup> For DASD MANAGER PLUS, the value is the database name.
**Table 25  BMCSYNC table (part 2 of 3)**

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
</table>
| UTILNAME      | CHAR(8)   | name of the executing utility:  
|               |           | - CHECK  
|               |           | - COPY  
|               |           | - STATS  
|               |           | - LOAD  
|               |           | - RECOVER  
|               |           | - REORG  
|               |           | - UNLOAD  |
| SHRLEVEL      | CHAR(1)   | degree to which utilities can share this object:  
|               |           | - Blank means that no status is requested, and any other utility can obtain any status.  
|               |           | - S allows sharing among any number of SHRLEVEL S utilities.  
|               |           | - X indicates that exclusive control is required. No other utility can run with SHRLEVEL X.  
|               |           | For more information, see Table 4 on page 87. |
| STATUS        | CHAR(1)   | status of the utility or object:  
|               |           | - blank (indicates no processing has been done)  
|               |           | - C (for CHECK PLUS, indicates checked)  
|               |           | - L (for LOADPLUS, indicates loaded)  
|               |           | - U (for UNLOAD PLUS, indicates unloaded)  
|               |           | - R (for REORG PLUS, indicates reloaded)  
|               |           | DASD MANAGER PLUS does not use this column. |
| XCOUNT        | INTEGER   | number of rows or keys processed in the current phase  
|               |           | DASD MANAGER PLUS does not use this column. |
| DDNAME        | CHAR(8)   | check, load, unload, or work ddname  
|               |           | DASD MANAGER PLUS does not use this column. |
| BLOCKS        | INTEGER   | number of blocks for the check, load, unload, or work data set  
|               |           | DASD MANAGER PLUS does not use this column. |
| ORIG_STATUS   | CHAR(8)   | encoded representation of the original DB2 status of the space  
|               |           | For RECOVER PLUS, this column restores the DB2 status of a space after recovery, if necessary.  
|               |           | DASD MANAGER PLUS does not use this column. |
| EXTRBA        | CHAR(6)   | (RECOVER PLUS) log point at which this space was externalized  
|               |           | RECOVER PLUS serialization logic uses this column. The other utilities do not use this column. |
Executing BMC utilities concurrently

BMC utility jobs register DB2 objects in the BMCSYNC table. The registering utility assigns a sharing level to each registered object. The sharing level controls access to that object from other BMC utilities. For partitioned DB2 spaces, registration is performed at the partition level.

The BMCSYNC table allows multiple BMC utilities (or multiple instances of a utility) to operate concurrently on different partitions of a DB2 space if no nonpartitioning indexes are involved. In addition, some BMC utilities can operate concurrently on the same object or partition. For information about which products can operate concurrently, see Table 26. For additional serialization and concurrency issues for each utility, see “Concurrency with other BMC utilities for DB2” on page 87.

The “Access level” column in Table 26 refers to the value of the “SHRLEVEL” column name in Table 25 on page 573.

Table 26  Executing BMC utilities concurrently (part 1 of 2)

<table>
<thead>
<tr>
<th>Product</th>
<th>Access level</th>
<th>Additional information</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHECK PLUS</td>
<td>S</td>
<td>none</td>
</tr>
<tr>
<td>COPY PLUS</td>
<td>S or blank</td>
<td>If you specify COPY IMAGECOPY, COPY PLUS registers the object with no access status (blank). Otherwise, COPY PLUS registers the object with shared access (S).</td>
</tr>
<tr>
<td>DASD MANAGER PLUS (BMCSTATS)</td>
<td>S</td>
<td>none</td>
</tr>
</tbody>
</table>
## Executing BMC utilities concurrently (part 2 of 2)

<table>
<thead>
<tr>
<th>Product</th>
<th>Access level</th>
<th>Additional information</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOADPLUS</td>
<td>X</td>
<td>If you specify PART, LOADPLUS registers only the specified partitions with exclusive access (X). If no nonpartitioned indexes exist on the table space, you can run other utilities on different partitions while running this job.</td>
</tr>
<tr>
<td>RECOVER PLUS</td>
<td>X, S, or blank</td>
<td>RECOVER PLUS registers an object with shared access (S) under the following conditions:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ The table space for an index is registered with shared access if the index is being rebuilt and its table space is not recovered in the same job.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ A table space partition is registered with shared access if the keys for that partition are unloaded with a RECOVER UNLOADKEYS operation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVER PLUS registers an object with no access status (blank) if you specify the following commands or options:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— the ACCUM command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— OUTCOPY ONLY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— INDEP OUTSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVER PLUS registers the object with exclusive access (X) in all other cases.</td>
</tr>
<tr>
<td>RECOVERY MANAGER</td>
<td>S</td>
<td>none</td>
</tr>
<tr>
<td>REORG PLUS</td>
<td>X</td>
<td>If you specify PART, REORG PLUS registers only the specified partitions with exclusive access (X). If no nonpartitioned indexes exist on the table space, you can run other utilities on different partitions while running this job.</td>
</tr>
<tr>
<td>UNLOAD PLUS</td>
<td>S</td>
<td>none</td>
</tr>
</tbody>
</table>
Considerations

Note the following considerations:

- You might need to increase the size of the BMCSYNC table space from the standard size that was allocated during installation when any of the following conditions exist:
  - You are processing a large number of partitions.

  Estimate this allocation based on the following factors:
  - number of utilities that you are executing concurrently
  - number of partitions that you are processing concurrently
  - number of files that you are allocating dynamically

- Do not run an IBM utility that attempts to manipulate data within the same objects on which a BMC utility is currently processing.

- If BMCSTATS is processing multiple objects and encounters an object that is held by another utility, the BMCSTATS job issues a warning. The warning identifies the object and the utility that is using it. BMCSTATS continues processing the next object.

- If BMCSTATS is processing an object and another utility requires exclusive control of that object, the other utility stops execution at initialization time.

Maintaining the BMCSYNC table

When a utility abends, rows might remain in the BMCSYNC table. If you need to control expansion of this table, use one of the following methods to delete rows:

- Use the TERM restart parameter on the EXEC statement to delete rows from the BMCUTIL and BMCSYNC tables. Do not delete any rows for instances of utilities that are awaiting restart.

- Delete invalid rows in the BMCUTIL table. Do not delete any rows for instances of utilities that are awaiting restart.
Cleaning up RECOVER UNLOADKEYS entries

Successful completion of a RECOVER UNLOADKEYS job leaves rows in BMCSYNC with blank utility IDs for table space partitions and indexes related to the unloaded keys. The table space rows prevent other BMC utilities from obtaining exclusive control of the table space. Running a RECOVER BUILDINDEX job removes these rows. Otherwise, you can remove them by running a job that uses the following statement for the table space and each index:

```
DELETE FROM creatorName.CMN_BMCSYNC
WHERE UTILID NOT IN
  (SELECT UTILID FROM creatorName.CMN_BMCUTIL);
```

**NOTE**
The names of the BMCUTIL and BMCSYNC tables might have been changed at your site during installation.

BMCTRANS table

Table 27 describes the contents of the BMCTRANS table, which contains information that RECOVERY MANAGER and Log Master use for transaction recovery. The table contains one row for each execution of Log Master (that is, one row for each log scan performed).

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>USERID</td>
<td>CHAR(8) NOT NULL</td>
<td>transaction creator</td>
</tr>
<tr>
<td>TRANID</td>
<td>VARCHAR(18) NOT NULL</td>
<td>transaction ID</td>
</tr>
<tr>
<td>STARTTIME</td>
<td>TIMESTAMP NOT NULL WITH DEFAULT</td>
<td>transaction start time</td>
</tr>
</tbody>
</table>
### Table 27  BMCTRANS table (part 2 of 3)

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PITRBA</td>
<td>CHAR(6) NOT NULL FOR BIT DATA</td>
<td>RBA for point-in-time recovery</td>
</tr>
<tr>
<td>OUTDSNAME</td>
<td>VARCHAR(35) NOT NULL</td>
<td>output data set prefix for SQL statements or the logical log</td>
</tr>
<tr>
<td>STATEa</td>
<td>SMALLINT NOT NULL</td>
<td>level of recovery analysis performed</td>
</tr>
<tr>
<td>PITTIME</td>
<td>TIMESTAMP NOT NULL WITH DEFAULT</td>
<td>timestamp for the PIT RBA</td>
</tr>
<tr>
<td>SEQNO</td>
<td>SMALLINT NOT NULL</td>
<td>sequence number of the filter text</td>
</tr>
<tr>
<td>PITWKEST</td>
<td>FLOAT NOT NULL</td>
<td>work estimate</td>
</tr>
<tr>
<td>FILTERLINE</td>
<td>VARCHAR(1040) NOT NULL</td>
<td>text of the filter (may span more than one row)</td>
</tr>
<tr>
<td>UNDONUMROWSUPD</td>
<td>FLOAT</td>
<td>number of unique rows (RIDs) that are selected by the filter of the log scan</td>
</tr>
<tr>
<td>UNDOSUBSEQUPDROWS</td>
<td>FLOAT</td>
<td>total number of anomaly log records relating to one of the rows (RIDs) selected by the log scan</td>
</tr>
<tr>
<td>UNDOLOGRECROWS</td>
<td>FLOAT</td>
<td>number of unique rows (RIDs) that are affected by an anomaly log record</td>
</tr>
<tr>
<td>UNDOJOBSTATUS</td>
<td>SMALLINT</td>
<td>code indicating the status of an UNDO log scan:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 (no action taken)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 (Log Master execution started)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 (Log Master execution completed successfully with return code 0,4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 (Log Master execution completed unsuccessfully with return code 8,12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 (Log Master execution abnormally ended)</td>
</tr>
<tr>
<td>REDOJOBSTATUS</td>
<td>SMALLINT</td>
<td>code indicating the status of a REDO log scan:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 (no action taken)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 (Log Master execution started)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 (Log Master execution completed successfully with return code 0,4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 (Log Master execution completed unsuccessfully with return code 8,12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 (Log Master execution abnormally ended)</td>
</tr>
</tbody>
</table>
BMCUTIL table

Table 27  BMCTRANS table  (part 3 of 3)

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENDTIME</td>
<td>TIMESTAMP NOT NULL WITH DEFAULT</td>
<td>transaction end time</td>
</tr>
<tr>
<td>ACTION</td>
<td>SMALLINT</td>
<td>code indicating what recovery, if any, has been performed on the transaction</td>
</tr>
</tbody>
</table>

If STATE equals 0, only UNDO analysis has been performed. If STATE is between 1 and 9999, UNDO and PIT analysis have been performed. If STATE is greater than 10000, UNDO, PIT, and REDO analysis have been performed.

BMCUTIL table

Table 28 describes the BMCUTIL table, which contains information about utilities that are currently running or started. The utilities use the table to control the use of utility IDs. Each BMC utility must have a unique ID for restart purposes. If you have more than one BMC utility installed, all of these utilities should share the same BMCUTIL table.

The utilities insert rows into the BMCUTIL table during the UTILINIT phase and update the table as the job status changes. The utilities delete rows from the BMCUTIL table during the UTILTERM phase.

Table 28  BMCUTIL table  (part 1 of 3)

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTILID</td>
<td>CHAR(16)</td>
<td>utility identifier</td>
</tr>
<tr>
<td>STATUS</td>
<td>CHAR(1)</td>
<td>execution status of the utility:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A (active, not executing command)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• I (initializing)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• P (pausing or pause-stopped)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• S (stopped)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• T (terminating)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• X (executing command)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DASD MANAGER PLUS uses only X.</td>
</tr>
<tr>
<td>UTILNAME</td>
<td>CHAR(8)</td>
<td>name of the executing utility:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CHECK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• COPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• STATS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LOAD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RECOVER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• REORG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• UNLOAD</td>
</tr>
</tbody>
</table>
### Table 28  BMCUTIL table (part 2 of 3)

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
</table>
| PHASE       | CHAR(8)   | current phase of the utility  
 COPY PLUS does not use this column. |
| USERID      | CHAR(8)   | user ID executing the utility |
| SSID        | CHAR(4)   | DB2 subsystem where the utility is running |
| RESTART     | CHAR(1)   | restart option:  
  - N (not restart)  
  - P (RESTART(PHASE))  
  - Y (RESTART)  
 DASD MANAGER PLUS does not use this column.  
 Although UNLOAD PLUS accepts the RESTART, RESTART(PHASE), NEW/RESTART, and NEW/RESTART(PHASE) parameters, the utility executes as though you had specified the NEW parameter. |
| NOTEID      | CHAR(8)   | TSO user ID to be notified  
 DASD MANAGER PLUS does not use this column. |
| DBNAME      | CHAR(8)   | (RECOVER PLUS and REORG PLUS) name of the database containing the table or index space for which the last checkpoint was taken  
 This value can be blank.  
 The other utilities do not use this column. |
| SPNAME      | CHAR(8)   | (RECOVER PLUS and REORG PLUS) name of the table or index space for which the last checkpoint was taken  
 This value can be blank.  
 The other utilities do not use this column. |
| SPSTATUS    | CHAR(5)   | (REORG PLUS) space status before the utility stopped  
 The other utilities do not use this column. |
| COMMANDNO   | SMALLINT  | not used (always 0) |
| COMMAND     | VARCHAR(256) | first 256 characters of the utility command text  
 RECOVER PLUS, DASD MANAGER PLUS, and COPY PLUS do not use this column. |
Maintaining the BMCUTIL table

When a utility abends, rows might remain in the BMCUTIL table. If you need to control expansion of this table, use one of the following methods to delete rows:

- Use the TERM restart parameter on the EXEC statement to delete rows from the BMCUTIL and BMCSYNC tables. Do not delete any rows for instances of utilities that are awaiting restart.

- Delete invalid rows in the BMCUTIL table. Do not delete any rows for instances of utilities that are awaiting restart.

Then use the following SQL statement to delete rows from the BMCSYNC table.

```sql
DELETE
FROM creatorName.CMN_BMCSYNC
WHERE UTILID NOT IN
  (SELECT UTILID FROM creatorName.CMN_BMCUTIL);
```

**NOTE**
The names of the BMCUTIL and BMCSYNC tables might have been changed at your site during installation.

### BMCXCOPY table

Table 29 on page 583 describes the contents of the BMCXCOPY table, which the BMC utilities use for tracking the following types of registered copies:

- indexes that COPY PLUS has copied:
  - COPY NO index copies
  - DSNUM \( n \) index (nonpartitioned) copies
  - incremental index copies
  - index copies that are made at data set level
- Instant Snapshots made by COPY PLUS with the BMC EXTENDED BUFFER MANAGER (XBM) product or BMC SNAPSHOT UPGRADE FEATURE (SUF) technology, and any standard copies made in association with the Instant Snapshot
- cabinet copies
- encrypted copies

The BMCXCOPY table functions like SYSIBM.SYSCOPY except that IXNAME replaces TSNAME in BMCXCOPY. You must control authorization and access to users for BMCXCOPY through standard DB2 authorization.

If you have more than one BMC utility installed, all of these utilities should share the same BMCXCOPY table.

### Table 29 BMCXCOPY table (part 1 of 4)

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBNAME</td>
<td>CHAR(8)</td>
<td>name of the database</td>
</tr>
<tr>
<td>IXNAME</td>
<td>CHAR(8)</td>
<td>name of the index space or table space for Instant Snapshots and associated copies</td>
</tr>
<tr>
<td>DSNUM</td>
<td>INTEGER</td>
<td>data set number within the index or table space</td>
</tr>
<tr>
<td>ICTYPE</td>
<td>CHAR(1)</td>
<td>operation type:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- F (COPY FULL YES; for COPY PLUS version 8.1 and later, Online Consistent Copies)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- I (COPY FULL NO)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- W (REORG LOG NO)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- X (REORG LOG YES)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- B (REBUILD INDEX)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- P (POINT-IN-TIME RECOVERY)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- C (for COPY PLUS version 7.3 and earlier, Online Consistent Copies)</td>
</tr>
<tr>
<td>ICDATE</td>
<td>CHAR(6)</td>
<td>date of the entry (ymmddd) for ICTYPE C, the consistent log point for the copy</td>
</tr>
<tr>
<td>START_RBA</td>
<td>CHAR(6)</td>
<td>a 48-bit positive integer containing the relative byte location of a point in the DB2 recovery log</td>
</tr>
</tbody>
</table>

The indicated point as follows:

- for ICTYPE F, the starting point for all updates since the image copy was taken
- for COPY_TYPE O, the minimum of the consistent point and the oldest inflight URID
- *(RECOVERY MANAGER)* for ICTYPE C, the consistent log point for the copy
  - RBA for non-data-sharing systems
  - LRSN for data sharing systems
### Table 29  BMCXCOPY table (part 2 of 4)

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILESEQNO</td>
<td>INTEGER</td>
<td>tape file sequence number of the copy</td>
</tr>
<tr>
<td>DEVTYPE</td>
<td>CHAR(8)</td>
<td>type of device on which the copy resides</td>
</tr>
<tr>
<td>IBMREQD</td>
<td>CHAR(1)</td>
<td>whether the row came from the basic machine-readable material (MRM) tape:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- N (NO)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Y (YES)</td>
</tr>
<tr>
<td>DSNAME</td>
<td>CHAR(44)</td>
<td>name of the data set</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If STYPE V, DSNAME is the name of the VSAM data component.</td>
</tr>
<tr>
<td>ICTIME</td>
<td>CHAR(6)</td>
<td>time at which this row was inserted (hhmmss)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The insertion takes place after the completion of the operation that the row represents.</td>
</tr>
<tr>
<td>SHRLEVEL</td>
<td>CHAR(1)</td>
<td>SHRLEVEL parameter on COPY if ICTYPE F:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- C (change)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- R (reference)</td>
</tr>
<tr>
<td>DSVOLSER</td>
<td>VARCHAR(1784)</td>
<td>volume serial numbers of the data set</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Commas separate items in a list of 6-byte numbers. This column is blank if the data set is cataloged.</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>TIMESTAMP</td>
<td>date and time when the row was inserted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This is the date and time that are recorded in ICDATE and ICTIME. The use of TIMESTAMP over ICDATE and ICTIME is recommended, because later DB2 releases might not support the latter two columns.</td>
</tr>
<tr>
<td>ICBACKUP</td>
<td>CHAR(2)</td>
<td>type of image copy contained in the data set:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- LB (data set contains local backup data)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- RP (data set contains recovery system main data)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- RB (data set contains recovery system backup data)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- blank (data set contains local system main data or is not one of multiple copies)</td>
</tr>
<tr>
<td>ICUNIT</td>
<td>CHAR(1)</td>
<td>media on which the image copy data set is stored:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- D (DASD)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- T (tape)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- blank (medium is neither tape nor DASD)</td>
</tr>
<tr>
<td>STYPE</td>
<td>CHAR(1)</td>
<td>type of copy:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- blank (for ICTYPE=F)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- V (Instant Snapshot or a VSAM data set)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- e (encrypted copy)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- B (when ICTYPE=P, BACKOUT recovery)</td>
</tr>
</tbody>
</table>
### Table 29  BMCXCOPY table (part 3 of 4)

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIT_RBA</td>
<td>CHAR(6)</td>
<td>point-in-time recovery:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- X’000000000000’ (for ICTYPE=F)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- consistent point (for COPY_TYPE=O)</td>
</tr>
<tr>
<td>GROUP_MEMBER</td>
<td>CHAR(8)</td>
<td>data-sharing group member (the name of the SSID where the copy was made)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This column is blank if you are not using data sharing.</td>
</tr>
<tr>
<td>OTYPE</td>
<td>CHAR(1)</td>
<td>type of object:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- T (table)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- I (index)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- i (compressed index)</td>
</tr>
<tr>
<td>LOWDSNUM</td>
<td>INTEGER</td>
<td>not used</td>
</tr>
<tr>
<td>HIGHDSNUM</td>
<td>INTEGER</td>
<td>not used</td>
</tr>
<tr>
<td>COPYPAGESF</td>
<td>FLOAT(8)</td>
<td>number of pages written to the copy data set</td>
</tr>
<tr>
<td>NPAGESF</td>
<td>FLOAT(8)</td>
<td>high-used RBA divided by the page size</td>
</tr>
<tr>
<td>CPAGESF</td>
<td>FLOAT(8)</td>
<td>total number of changed pages</td>
</tr>
<tr>
<td>JOBNAME</td>
<td>CHAR(8)</td>
<td>job name</td>
</tr>
<tr>
<td>AUTHID</td>
<td>CHAR(8)</td>
<td>authorization ID</td>
</tr>
<tr>
<td>OLDEST_VERSION</td>
<td>SMALLINT</td>
<td>when ICTYPE= B, F, I, S, W, or X, the version number of the oldest format of data for an object</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For other values of ICTYPE, the value is –1.</td>
</tr>
<tr>
<td>LOGICAL_PART</td>
<td>INTEGER</td>
<td>logical partition number</td>
</tr>
<tr>
<td>LOGGED</td>
<td>CHAR(1)</td>
<td>logging attribute of the table space:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Y (logged)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- N (not logged)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- blank (row inserted prior to DB2 Version 9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For a non-LOB table space or index space, blank indicates that the logging attribute is logged.</td>
</tr>
<tr>
<td>TTYPE</td>
<td>CHAR(8)</td>
<td>row format for the table space or partition:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- RRF (reordered row format)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- BRF (basic row format)</td>
</tr>
<tr>
<td>INSTANCE</td>
<td>SMALLINT</td>
<td>instance number of the current base objects (table and index)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 1.</td>
</tr>
<tr>
<td>RELCREATED</td>
<td>CHAR(1)</td>
<td>DB2 release that created the object</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the release is earlier than DB2 Version 9, the value is blank.</td>
</tr>
</tbody>
</table>
Maintaining the BMCXCOPY table

Periodically, you should review BMCXCOPY and delete old rows to control its expansion. To delete all rows from the BMCXCOPY table that are older than 30 days, use the following statement as an example:

```
DELETE
FROM creatorName.CMN_BMCXCOPY
WHERE DAYS(CURRENT_TIMESTAMP) - DAYS(TIMESTAMP) > 30;
```
BMC Common DB2 repository

This appendix presents the following topics:

BMC Common DB2 repository tables ................................................................. 587
  Naming conventions ................................................................................. 587
  Object set table ................................................................................. 588
  Object set definition table ................................................................. 589
  Object set SQL table ........................................................................ 590
  Group options table .......................................................................... 590
  Product registration table ................................................................. 591
  Group authorizations table ............................................................... 591

BMC Common DB2 repository tables

The DB2 tables that compose the BMC Common DB2 repository are described in the following sections.

Naming conventions

This section describes the naming conventions for BMC Common DB2 repository tables. Table 30 on page 588 provides the synonyms and local table names.

--- NOTE ---

Note that synonyms cannot be different and tables names may be different at your site based upon options chosen during product installation.
Object set table

Table 30  BMC Common DB2 repository synonym and local table names

<table>
<thead>
<tr>
<th>Synonym</th>
<th>Local table name</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMCSCC_OBJSETS</td>
<td>BMCUTIL.CMN_OS</td>
</tr>
<tr>
<td>BMCSCC_OBJSET_DEF</td>
<td>BMCUTIL.CMN_OS_DEF</td>
</tr>
<tr>
<td>BMCSCC_OBJSET_SQL</td>
<td>BMCUTIL.CMN_OS_SQL</td>
</tr>
<tr>
<td>BMCSCC_GRPPOPTS</td>
<td>BMCUTIL.CMN_OS_OPTS</td>
</tr>
<tr>
<td>BMCSCC_PRODREG</td>
<td>BMCUTIL.CMN_OS_PREG</td>
</tr>
<tr>
<td>BMCSCC_GROUPAUTH</td>
<td>BMCUTIL.CMN_OS_GAUTH</td>
</tr>
</tbody>
</table>

Table 31 describes the contents of the OBJSETS table. This table describes and provides information about object sets. This table contains one row for each object set defined in the repository.

Table 31  OBJSETS table

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSNAME</td>
<td>VARCHAR(27) NOT NULL</td>
<td>name of object set</td>
</tr>
<tr>
<td>CREATE_TSMP</td>
<td>TIMESTAMP NOT NULL WITH DEFAULT</td>
<td>timestamp of object set creation</td>
</tr>
<tr>
<td>CREATE_UID</td>
<td>CHAR(8) NOT NULL</td>
<td>AUTHID of creator of the object set</td>
</tr>
<tr>
<td>UPDATE_TSMP</td>
<td>TIMESTAMP NOT NULL WITH DEFAULT</td>
<td>timestamp of last maintenance activity</td>
</tr>
<tr>
<td>UPDATE_UID</td>
<td>CHAR(8) NOT NULL</td>
<td>AUTHID of last updater of the object set</td>
</tr>
<tr>
<td>DESCRIPTION</td>
<td>VARCHAR(60) NOT NULL</td>
<td>description of the object set</td>
</tr>
<tr>
<td>PRODUCT_ID</td>
<td>CHAR(3) NOT NULL</td>
<td>creating product ID</td>
</tr>
<tr>
<td>TYPE</td>
<td>CHAR(2) NOT NULL</td>
<td>product group type</td>
</tr>
<tr>
<td>NUMBER_OBJECTS</td>
<td>INTEGER NOT NULL WITH DEFAULT</td>
<td>number of objects from last open</td>
</tr>
<tr>
<td>CHECKSUM</td>
<td>SMALLINT NOT NULL</td>
<td>verification value from API updates</td>
</tr>
</tbody>
</table>

Recover Plus (COPY PLUS)

Arm (RECOVERY MANAGER)
Table 32 describes the contents of the OBJSET_DEF table. This table contains one row for each object set definition specification defined for an object set.

### Table 32  OBJSET_DEF table (part 1 of 2)

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Description of Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSNAME</td>
<td>VARCHAR(27)</td>
<td>name of the object set</td>
</tr>
<tr>
<td>SEQNO</td>
<td>SMALLINT</td>
<td>sequence number of definition</td>
</tr>
<tr>
<td>INCEXC_IND</td>
<td>CHAR(1)</td>
<td>include or exclude indicator (+, -)</td>
</tr>
<tr>
<td>PATTERN_TYPE</td>
<td>CHAR(2)</td>
<td>Pattern for include or exclude:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ TS (table space name pattern)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ IX (index name pattern)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ TB (table name pattern)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ IS (index space name pattern)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ PL (plan name pattern)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ PG (package name pattern)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ SG (stogroup name pattern)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ OS (object set name pattern)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ SQ (dynamic SQL pattern)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ RP (repository plan)</td>
</tr>
<tr>
<td>INC_IX</td>
<td>CHAR(1)</td>
<td>include related indexes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Y (Yes)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ N (No)</td>
</tr>
<tr>
<td>INC_RI</td>
<td>CHAR(1)</td>
<td>include RI objects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Y (Yes)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ N (No)</td>
</tr>
<tr>
<td>INC_LOBS</td>
<td>CHAR(1)</td>
<td>include LOB objects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Y (Yes)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ N (No)</td>
</tr>
<tr>
<td>INC_XML</td>
<td>CHAR(1)</td>
<td>include XML objects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Y (Yes)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ N (No)</td>
</tr>
<tr>
<td>INC_CLONES</td>
<td>CHAR(1)</td>
<td>include clones only</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Y (Yes)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ N (No)</td>
</tr>
<tr>
<td>BY_PART</td>
<td>CHAR(1)</td>
<td>expand objects by partition</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Y (Yes)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ N (No)</td>
</tr>
<tr>
<td>PART_BEG</td>
<td>SMALLINT</td>
<td>beginning partition number (0-4096)</td>
</tr>
<tr>
<td>PART_END</td>
<td>SMALLINT</td>
<td>ending partition number (0-4096)</td>
</tr>
</tbody>
</table>
Table 32  OBJSET_DEF table  (part 2 of 2)

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Description of Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>REF_SEQ_NBR</td>
<td>SMALLINT NOT NULL</td>
<td>for future use</td>
</tr>
<tr>
<td>DESC</td>
<td>VARCHAR(60) NOT NULL</td>
<td>description of the specification</td>
</tr>
<tr>
<td>OBJ_QUAL1</td>
<td>VARCHAR(128) NOT NULL</td>
<td>object qualifier 1</td>
</tr>
<tr>
<td>OBJ_QUAL2</td>
<td>VARCHAR(128) NOT NULL</td>
<td>object qualifier 2</td>
</tr>
<tr>
<td>OBJ_QUAL3</td>
<td>VARCHAR(128) NOT NULL</td>
<td>object qualifier 3</td>
</tr>
<tr>
<td>UNI_QUALS</td>
<td>CHAR(1) NOT NULL</td>
<td>UNICODE indicator</td>
</tr>
<tr>
<td>UPDATE_UID</td>
<td>CHAR(8) NOT NULL</td>
<td>ID of last updater of object set definitions</td>
</tr>
<tr>
<td>UPDATE_TSMP</td>
<td>TIMESTAMP NOT NULL WITH DEFAULT</td>
<td>timestamp of last maintenance activity</td>
</tr>
<tr>
<td>PACKAGE_VERSION</td>
<td>SMALLINT NOT NULL</td>
<td>package version</td>
</tr>
<tr>
<td>INC_HISTORY</td>
<td>CHAR(1) NOT NULL WITH DEFAULT 'N'</td>
<td>include related history objects</td>
</tr>
<tr>
<td></td>
<td>Y (Yes)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N (No)</td>
<td></td>
</tr>
</tbody>
</table>

Object set SQL table

Table 33 describes the contents of the OBJSET_SQL table. This table contains one row for each object set specification in dynamic SQL (type SQ).

Table 33  OBJSET_SQL table

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSNAME</td>
<td>VARCHAR(27) NOT NULL</td>
<td>name of the object set</td>
</tr>
<tr>
<td>SPEC_SEQNO</td>
<td>SMALLINT NOT NULL</td>
<td>sequence number from OBJSET_DEF table</td>
</tr>
<tr>
<td>SEQNO</td>
<td>SMALLINT NOT NULL</td>
<td>sequence number to order multiple SQL entries</td>
</tr>
<tr>
<td>TEXT</td>
<td>VARCHAR(72) NOT NULL</td>
<td>line of SQL text</td>
</tr>
</tbody>
</table>

Group options table

Table 34 on page 591 describes the contents of the GRPOPTS table. This table contains one row for each option defined to either a defined group, or a subsystem level option. For information about the recover and backup options supported by RECOVERY MANAGER, see the RECOVERY MANAGER for DB2 User Guide.
Product registration table

Table 35 describes the contents of the PRODREG table. There should be one entry for each product and version that is registered.

Table 35 PRODREG table

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRODUCT_ID</td>
<td>CHAR(3) NOT NULL</td>
<td>product ID</td>
</tr>
<tr>
<td>PLAN_NAME</td>
<td>VARCHAR(24) NOT NULL</td>
<td>plan name</td>
</tr>
<tr>
<td>PRODUCT_VERSION</td>
<td>CHAR(4) NOT NULL</td>
<td>product version</td>
</tr>
</tbody>
</table>

Group authorizations table

Table 36 describes the contents of the GROUPAUTH table. This table optionally contains one row for each authority granted on a group. No rows exist if no authority has been granted.

Table 36 GROUPAUTH table

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSNAME</td>
<td>VARCHAR(27) NOT NULL</td>
<td>name of object set</td>
</tr>
<tr>
<td>GRANTEE</td>
<td>CHAR(8) NOT NULL</td>
<td>AUTHID to whom authorization was granted</td>
</tr>
<tr>
<td>TYPE</td>
<td>CHAR(1) NOT NULL</td>
<td>type of authorization granted</td>
</tr>
<tr>
<td>GRANTOR</td>
<td>CHAR(8) NOT NULL</td>
<td>grantor of authorization</td>
</tr>
<tr>
<td>DATE_GRANTED</td>
<td>TIMESTAMP NOT NULL</td>
<td>WITH DEFAULT</td>
</tr>
</tbody>
</table>
For quick reference, this appendix provides the syntax diagrams for the RECOVER PLUS commands without any option descriptions. Cross-references to the option descriptions are included in the diagrams.

Alphabetical listing of RECOVER PLUS options .................................. 594
OPTIONS command syntax diagram .................................................. 601
OUTPUT command syntax diagram ................................................... 604
RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE command syntax diagram .................................................. 606
REBUILD INDEX command syntax diagram ....................................... 615
RECOVER UNLOADKEYS command syntax diagram ............................ 616
RECOVER BUILDINDEX command syntax diagram ............................... 617
LOGSCAN command syntax diagram ................................................. 618
RECOVER PLUS options are shown in Table 37, alphabetized by RECOVER PLUS command, and within the command by option name. The last column contains a page reference for each option.

### Table 37  RECOVER PLUS command options (part 1 of 7)

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGSCAN</td>
<td>244</td>
<td>AUX</td>
<td>250, 254</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BACKOUT</td>
<td>259</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
<td>255</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM</td>
<td>252</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMLOGPOINT</td>
<td>250</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTARCHQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTCOPY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTQUIESCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTSHUTDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• X’logPoint’</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMRBA</td>
<td>250</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTARCHQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTCOPY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTQUIESCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTSHUTDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• X’logPoint’</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEX</td>
<td>252</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• for single index specification</td>
<td>253</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• for multiple index specification</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXES (or INDEX)</td>
<td>249</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• for single index specification</td>
<td>252</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• for multiple index specification</td>
<td>254</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGAPPLY ONLY</td>
<td>260</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGONLY</td>
<td>260</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
<td>254</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
<td>248</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TOLOGPOINT</td>
<td>257</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTARCHQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTCOMMONQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTQUIESCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(relativeGenerationNumber)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTSHUTDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• X’logPoint’</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>TORBA</td>
<td>257</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTARCHQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTCOMMONQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTQUIESCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(relativeGenerationNumber)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTSHUTDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• X’logPoint’</td>
<td></td>
</tr>
<tr>
<td>Command name</td>
<td>Page</td>
<td>Command option</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------</td>
<td>---------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>OPTIONS or OPTION</td>
<td>109</td>
<td>ANALYZE</td>
<td>132</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTOSIZE</td>
<td>140</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUX</td>
<td>122</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BACKOUT</td>
<td>124</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CHECKINT</td>
<td>117</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DATAMVR</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EARLYCAT</td>
<td>112</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EARLYRECALL</td>
<td>112</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXLOG</td>
<td>126</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IXRECP</td>
<td>128</td>
</tr>
<tr>
<td></td>
<td></td>
<td>KEYSORT</td>
<td>131</td>
</tr>
<tr>
<td></td>
<td></td>
<td>KSORTSHARE</td>
<td>139</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGPOINT X‘logPoint’ or LOGMARK</td>
<td>143</td>
</tr>
<tr>
<td></td>
<td></td>
<td>logMarkName (logMarkGeneration)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGSORT</td>
<td>113</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXDRIVES</td>
<td>120</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXKSORT</td>
<td>136</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXLOGS</td>
<td>121</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXLSORT</td>
<td>138</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOEARLYCAT</td>
<td>112</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOEARLYRECALL</td>
<td>112</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSYSLGRNG</td>
<td>116</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NUMREC</td>
<td>115</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ABS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>AVGRECSZ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CALC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>EST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOEST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ON ERROR ANY CONTINUE</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPY</td>
<td>121</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ASCODED</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>BYPART</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYPOINTER</td>
<td>140</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESINV</td>
<td>119</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESOURCE SELECTION</td>
<td>120</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ACCUMS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>COPIES</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SIMULATE YES</td>
<td>134</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SMCORE</td>
<td>119</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT for KEYSORT</td>
<td>131</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for LOGSORT</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDIAG</td>
<td>118</td>
</tr>
</tbody>
</table>
Alphabetical listing of RECOVER PLUS options

Table 37 RECOVER PLUS command options (part 3 of 7)

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTIONS or OPTION (continued)</td>
<td>109</td>
<td>SORTDYN</td>
<td>119</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for KEYSORT</td>
<td>132</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for LOGSORT</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STOGROUP... USEORDER</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>TIMESTAMP</td>
<td>143</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRTCH</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td></td>
<td>URIDDDN</td>
<td>142</td>
</tr>
<tr>
<td></td>
<td></td>
<td>USEACCUM</td>
<td>116</td>
</tr>
<tr>
<td></td>
<td></td>
<td>USEHROBIDS</td>
<td>123</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WTOR</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XBMID</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZIIP</td>
<td>130</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>145</td>
<td>CATLG</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DATACLAS</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNAME</td>
<td>147</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EATTR</td>
<td>151</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXPDT</td>
<td>154</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MGMTCLAS</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODELDCB</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RETPD</td>
<td>153</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPACE</td>
<td>151</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STACK</td>
<td>152</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STORCLAS</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRTCH</td>
<td>153</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNIT</td>
<td>148</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNITCNT</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VOLUMES</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VOLUMES</td>
<td>151</td>
</tr>
</tbody>
</table>
Table 37   RECOVER PLUS command options (part 4 of 7)

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>REBUILD INDEX</td>
<td>212</td>
<td>ANALYZE</td>
<td>225</td>
</tr>
<tr>
<td>SIMRBLD INDEX</td>
<td></td>
<td>AUX</td>
<td>219</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
<td>221</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP INTABLESPACE</td>
<td>219</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP OUTSPACE</td>
<td>217</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for INDEP INTABLESPACE</td>
<td>218</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSCRATCH</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NUMREC</td>
<td>224</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ABS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CALC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>EST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOEST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOWORKDDN</td>
<td>222</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
<td>216</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PART</td>
<td>220</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDEFINE</td>
<td>226</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REUSE</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT</td>
<td>222</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
<td>222</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
<td>219</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WORKDDN</td>
<td>221</td>
</tr>
<tr>
<td>RECOVER BUILDINDEX</td>
<td>237</td>
<td>ANALYZE</td>
<td>241</td>
</tr>
<tr>
<td>SIMRCVR BUILDINDEX</td>
<td></td>
<td>CLONE</td>
<td>241</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP OUTSPACE</td>
<td>239</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
<td>240</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSCRATCH</td>
<td>243</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDEFINE</td>
<td>242</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REUSE</td>
<td>243</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEY</td>
<td>241</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEYDDN</td>
<td>241</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
<td>240</td>
</tr>
</tbody>
</table>
Table 37  RECOVER PLUS command options (part 5 of 7)

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX</td>
<td>155</td>
<td>ANALYZE</td>
<td>189</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>AUX</td>
<td>169, 180</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>BACKOUT</td>
<td>184</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>CLONE</td>
<td>189</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>DBID</td>
<td>194</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>DROPRECOVERY</td>
<td>170</td>
</tr>
<tr>
<td>DSNUM</td>
<td></td>
<td>DSSIZE</td>
<td>205</td>
</tr>
<tr>
<td>for INDEX</td>
<td></td>
<td>ENCRYPTED</td>
<td>206</td>
</tr>
<tr>
<td>for INDEXSPACE</td>
<td></td>
<td>FROMLOGPOINT</td>
<td>198</td>
</tr>
<tr>
<td>for TABLESPACE</td>
<td></td>
<td>LASTARCHQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTCOPY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTQUIESCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTSHUTDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'\logPoint'</td>
<td></td>
</tr>
<tr>
<td>FROMRBA</td>
<td></td>
<td>INCOPY FULL</td>
<td>195</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INCR</td>
<td>196</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP OUTSPACE</td>
<td>178</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEVT</td>
<td>207</td>
</tr>
<tr>
<td>INDEX</td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td>for single index specification</td>
<td></td>
<td>INDEXES (or INDEX)</td>
<td>168</td>
</tr>
<tr>
<td>for multiple index specification</td>
<td></td>
<td>INDEXSPACE</td>
<td></td>
</tr>
<tr>
<td>for single index specification</td>
<td></td>
<td>INDEXSPACE</td>
<td>172</td>
</tr>
<tr>
<td>for multiple index specification</td>
<td></td>
<td>INDEXSPACE</td>
<td>174</td>
</tr>
<tr>
<td>INLINE</td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td>INLOG LOGPOINT</td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td>INLOG RBA</td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td>INSEQNO</td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td>INVOLUME</td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td>LASTCOPY</td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td>LASTSITE</td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td>LOGAPPLY ONLY</td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td>LOGONLY</td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td>LOGPOINT</td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEX</td>
<td></td>
</tr>
</tbody>
</table>
### Table 37  RECOVER PLUS command options (part 6 of 7)

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX</td>
<td>155</td>
<td>LOGSORT</td>
<td>188</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>MODEL</td>
<td>180</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>NOCOPYPEND</td>
<td>197</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>NOSCATCH</td>
<td>191</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>NOWORKDDN</td>
<td>173, 175</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>NUMREC</td>
<td>115</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ABS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>AVGRECSZ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CALC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>EST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOEST</td>
<td></td>
</tr>
<tr>
<td>OBID</td>
<td>195</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OBIDXLAT</td>
<td>193</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OBJECTSET</td>
<td>176</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUTCOPY</td>
<td>199</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUTCOPYDDN</td>
<td>208</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUTCOPYDSN</td>
<td>208</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSID</td>
<td>195</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIECESIZE</td>
<td>205</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RBA</td>
<td>206</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECOVERYDDN</td>
<td>209</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECOVERYDSN</td>
<td>209</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECOVERYSITE</td>
<td>189</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REDEFINE</td>
<td>191</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REGISTER</td>
<td>202</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESET</td>
<td>194</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESETRTS</td>
<td>203</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REUSE</td>
<td>192</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHRLEVEL</td>
<td>206</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SNAPSHOT</td>
<td>196</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SORTEVT</td>
<td>114</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SORTNUM</td>
<td>114</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>204</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TABLESPACE</td>
<td></td>
<td>for table space specification</td>
<td>168</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for multiple index specification</td>
<td>174</td>
</tr>
<tr>
<td>TOCOPY (relativeGenerationNumber)</td>
<td>185</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOLOGPOINT</td>
<td>181</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTARCHQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTCOMMONQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTQUIESCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(relativeGenerationNumber)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTSHUTDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGMARK logMarkName</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(logMarkGeneration)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>X’logPoint”</td>
<td></td>
</tr>
</tbody>
</table>
### Table 37  RECOVER PLUS command options (part 7 of 7)

<table>
<thead>
<tr>
<th>Command name</th>
<th>Page</th>
<th>Command option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX</td>
<td>155</td>
<td>TOSEQNO</td>
<td>187</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>TOVOLUME</td>
<td>187</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>TRANSFORM</td>
<td>171</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>UPDATE VERSIONS</td>
<td>192</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(continued)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECOVER UNLOADKEYS</td>
<td>228</td>
<td>ANALYZE</td>
<td>235</td>
</tr>
<tr>
<td>SIMRCVR UNLOADKEYS</td>
<td></td>
<td>CLONE</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP INTABLESPACE</td>
<td>231</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NUMREC</td>
<td>234</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ABS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ CALC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ EST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ NOEST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>PART</td>
<td>230</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEY</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEDDN</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT</td>
<td>233</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
<td>233</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
<td>230</td>
</tr>
</tbody>
</table>
OPTIONS command syntax diagram

Figure 158 OPTIONS syntax diagram (part 1 of 2)

*the default for ANALYZE ONLY runs
OPTIONS, continued

- **BACKOUT**
  - AUTO**
  - NO
  - YES
  - page 124

- **INDEXLOG**
  - YES
  - NO
  - AUTO**
  - page 126

- **IXRECP**
  - NO
  - YES
  - page 128

- **TRTCH**
  - NONE
  - COMP
  - NOCOMP
  - page 129

- **XBMID ssid or xbmGroup**
  - page 129

- **SORTDEVT deviceType**
- **SORTNUM integer**
  - page 131

- **KEYSORT**
  - page 131

- **ANALYZE**
  - YES
  - NO
  - page 132

- **SIMULATE YES**
  - page 134

- **LOGPOINT X'logPoint'**
  - page 135

- **WTOR**
  - YES
  - NO
  - page 135

- **ON ERROR**
  - CONTINUE integer
  - page 135
  - ANY

- **MAXKSORT integer**
  - page 136

- **MAXLSORT integer**
  - page 138

- **KSORTSHARE**
  - YES
  - NO
  - page 139

- **DATAMVR programName**
  - page 131

- **RECOVERYPOINT**
  - ** page 140

- **TIMESTAMP yyyy-mm-dd-h.mm.ss.tttt**

- **URIDDN (output)**
  - page 142

- **LOGPOINT X'logPoint'
  - page 143

- **LOGMARK logMarkName
  - (logMarkGeneration)

*valid with R+/CHANGE ACCUM only
**valid only with a Recovery Management for DB2 solution password
OUTPUT command syntax diagram

Figure 162  OUTPUT syntax

Figure 163  OUTPUT syntax—Common options

Figure 164  OUTPUT syntax—Disk options
Figure 165 OUTPUT syntax—Tape options
Recover Tablespace, Recover Index, Recover Indexspace Syntax Diagram

Figure 166  Recover Tablespace, Recover Index, and Recover Indexspace Syntax Diagram

- Recover Tablespace
- Recover Index
- Recover Indexspace

Table Space Specification
- Single Index Specification
- Multiple Index Specification

Objectset creator.name

Logsort Specification
- Logapply only
- Logonly

Recovery Site
- Localsite
- Clone

Update Versions
- Yes
- No
- Only

Redefine
- Yes
- No
- Noscratch

*Not valid with BACKOUT
**For special consideration, see the descriptions on page 171 and page 174.
Figure 167  RECOVER syntax—Table space specification

```plaintext
Table space specification

TABLESPACE
  page 168
    tableSpaceName
      DSNDDB04
      databaseName.
      OBJECTSET
      creator.name
      page 168
      INDEXES
      INDEX
      page 168
      NO
      YES

DSNUM
  page 172
    ALL
    begin : end
    integer *
    page 169
    AUX
    NO
    YES
    XML
    LOB
    HISTORY

TRANSFORM
  page 171
  Non-registered copy or INLOG
  specification
  page 611

FROMRBA or FROMLOGPOINT
  specification
  page 612

OUTCOPY
  specification
  page 612

*Not valid with TRANSFORM
**Not valid with BACKOUT
```

Appendix E  RECOVER PLUS syntax diagrams  607
Figure 168  RECOVER syntax—Single index specification

*For special considerations, see the description on page 171.
** Valid with INDEXLOG AUTO.
Figure 169  RECOVER syntax—Multiple index specification

*Allowed with multiple recoveries only if dynamic outcopy allocation (OUTPUT syntax is used.
**For special considerations, see the description on page 174.
*** Valid with INDEXLOG AUTO.

Figure 170  RECOVER syntax—INDEPENDENT OUTSPACE specification

*Not valid with BACKOUT
Figure 174  RECOVER syntax—OBIDXLAT specification

**OBIDXLAT specification**

- OBIDXLAT*
  - RESET
  - DBID
    - \( X'\text{hexSourceID}' \)
    - decimalSourceID
  - \( X'\text{hexTargetID}' \)
    - decimalTargetID
  - \( X'\text{hexTargetID}' \)
    - decimalTargetID
- PSID
  - \( X'\text{hexSourceID}' \)
    - decimalSourceID
  - \( X'\text{hexTargetID}' \)
    - decimalTargetID
  - \( X'\text{hexTargetID}' \)
    - decimalTargetID
- OBID
  - \( X'\text{hexSourceID}' \)
    - decimalSourceIDD
  - \( X'\text{hexTargetID}' \)
    - decimalTargetID
  - \( X'\text{hexTargetID}' \)
    - decimalTargetID
  - \( X'\text{hexTargetID}' \)
    - decimalTargetID

*Not valid with BACKOUT

Figure 175  RECOVER syntax — Non-registered copy or INLOG specification

**Non-registered copy or INLOG specification**

- INCOPY
  - FULL
  - INLINE *
  - SNAPSHOT
    - page 196
  - INLOG RBA *,**
  - INLOG LOGPOINT *,**
    - X'logPoint *'
    - INCR*
  - NOCOPYPEND

*Not valid with index specification
**Not valid with BACKOUT
Figure 176  RECOVER syntax—FROMRBA or FROMLOGPOINT specification

Figure 177  RECOVER syntax—OUTCOPY specification

**Not valid with BACKOUT or Instant Snapshot copies**
Figure 178 RECOVER syntax—INCOPY specification

![INCOPY specification diagram]

- **DSNAME dataSetName**
  - page 204
- **MODEL dataSetName**
  - page 205
- **DSIZE (integerG)**
  - page 205
- **PIECESIZE (...)**
  - page 205
- **LOGPOINT**
  - page 206
- **X'logPoint'**
  - page 206
- **SHRLEVEL**
  - page 206
- **REFERENCE**
- **CHANGE**
- **INVENTORY**
  - page 207
- **CATALOG**
- **volumeSerialNumber**
- **INDEVT**
- **deviceType**
- **INSEQNO**
- **integer**
- **RBA**
- **ENCRIPTED**
- **TIMESTAMP**
  - timestamp
- **DSSIZE (integerG)**
- **MODEL dataSetName**
- **page 205**
- **PIECESIZE (...)**
- **integerK**
- **integerM**
- **integerG**
- **Valid only with INCOPY FULL SNAPSHOT**

Figure 179 RECOVER syntax—OUTCOPYDDN specification

![OUTCOPYDDN specification diagram]

- **OUTCOPYDDN**
  - page 208
- **BMCCPY**
  - ***
- **DDName1**
- **DDNamePrefix1**
- **outputDescriptor1***
- **BMCCPZ**
  - ***
- **DDName2**
- **DDNamePrefix2**
- **outputDescriptor2***
- **OUTCOPYDSN**
  - page 208
  - dataSetName1
  - dataSetName2
- **page 208**

*These default values apply to table spaces with fewer than 100 partitions. If the table space has 100 partitions or more, the corresponding default values are BMCCY and BMCCZ.

**These values are the default values that are used by the OUTPUT command.

***These values are used to specify the OUTPUT command, which will control the allocation of the copy.
**Figure 180  RECOVER syntax—RECOVERYDDN specification**

*These default values apply to table spaces with fewer than 100 partitions. If the table space has 100 partitions or more, the corresponding default values are BMCRY and BMCRZ.

**These values are the default values that are used by the OUTPUT command.

***These values are used to specify the OUTPUT command, which will control the allocation of the copy.
REBUILD INDEX command syntax diagram

Figure 181  REBUILD INDEX syntax diagram

- `REBUILD` INDEX
  - `RECOVER*`
  - `SIMRBLD`
  - `SIMRCVR***`

- `INDEP OUTSPACE`

- `INDEP INTABLESPACE`

- `TABLESPACE`
  - `ALL`
  - `PART begin : end`

- `OBJECTSET creator.name`

- `INDEP OUTSPACE`
  - `MODEL vcat.BMCDBC.databaseName.indexspaceName.I0001.znnn`
  - `MODEL userNamedDataSet`

- `TABLESPACE`
  - `DSNDB04` databaseName.
  - `tableSpaceName`

- `INDEP INTABLESPACE`
  - `MODEL vcat.BMCDBC.databaseName.tableSpaceName.I0001.znnn`
  - `MODEL userNamedDataSet`

- `PART integer`
  - `begin : end`

- `CLONE`

- `WORKDDN`
  - `SYSUT1`
  - `DDName`
  - `DDNamePrefix`

- `NOWORKDDN`

- `SORTDEVT deviceType`
  - `SORTNUM integer`
  - `NUMREC`
    - `CALC`
    - `NOEST`
    - `EST integer`
    - `ABS integer`

- `ANALYZE`
  - `YES`
  - `NO`
  - `ONLY`

- `REDEFINE`
  - `YES`
  - `NO`
  - `NOSCRATCH`

- `REUSE`

*Is synonymous to REBUILD when INDEXLOG=NO
**Is synonymous to SIMRBLD when INDEXLOG=NO
Figure 182  RECOVER UNLOADKEYS syntax diagram

*You cannot specify PART in both places.*
RECOVER BUILDINDEX command syntax diagram

Figure 183  RECOVER BUILDINDEX syntax diagram

```
RECOVER BUILDINDEX (indexName, ALL)

INDEP OUTSPACE

MODEL vcat.BMCDBC.databaseName.indexSpaceName.I0001.znnn
    userNamedDataSet

TABLESPACE
    DSNDB04.databaseName.

SKEYDDN
    SKEY DDNamePrefix

ANALYZE
    YES
    NO
    ONLY

REDEFINE
    YES
    NOSCRATCH
    NO

REUSE
```

Appendix E  RECOVER PLUS syntax diagrams  617
LOGSCAN command syntax diagram

Figure 184  LOGSCAN syntax diagram

Figure 185  LOGSCAN syntax—Table space specification
Figure 186  LOGSCAN syntax—Single index specification

Single index specification

Figure 187  LOGSCAN syntax—Multiple index specification

Multiple index specification
Figure 188  LOGSCAN syntax—Point-in-time recovery specification
Glossary

The terms in this glossary include acronyms that are used in this book. The definitions are based on those in current DB2 and operating system publications and in the IBM Dictionary of Computing.

A

abend

See abnormal end of task.

abend reason code

A 4-byte hexadecimal code that uniquely identifies a problem with DB2. A complete list of DB2 abend reason codes and their explanations is contained in DB2 for z/OS Codes and DB2 for z/OS Messages.

abnormal end of task

Termination of a task, a job, or a subsystem because of an error condition that cannot be resolved during execution by recovery facilities.

access method

A technique for moving data between main storage and input/output devices.

active log

The portion of the DB2 log to which log records are written as they are generated. The active log always contains the most recent log records, whereas the archive log holds those records that are older and no longer will fit on the active log. For recovery purposes, this is the log in use when a failure occurs.

address space

A range of virtual storage pages identified by a number (ASID) and a collection of segment and page tables which map the virtual pages to real pages of the computer’s memory.

agent

A submitted job or started task that communicates information about the DB2 subsystems on a particular system for SHRLEVEL CHANGE copy jobs in a data sharing environment. There must one agent per system with an active DB2 data sharing member. The agent is used to determine the registration point for copies.

APF

See authorized program facility.
API
   See application programming interface (API).

application
   A program or set of programs that perform a task (for example, a payroll application).

application program interface
   A functional interface supplied by the operating system or by a separately orderable licensed program that allows an application program written in a high-level language to use specific data or functions of the operating system or licensed program.

application plan
   In DB2, the control structure produced during the bind process and used to process SQL statements encountered during statement execution.

archive log
   The portion of the DB2 log that contains log records that have been copied from the active log. The archive log is retrievable in the event that it is needed for recovery purposes.

authorization ID
   A string that can be verified for connection to DB2 and to which a set of privileges are allowed. It can represent an individual, an organizational group, or a function, but DB2 does not determine this representation.

authorized program facility
   A facility that permits identification of programs authorized to use restricted functions.

auxiliary table
   A table that stores columns outside the table in which they are defined.

B

basic sequential access method
   An access method for storing or retrieving data blocks in a continuous sequence, by using either a sequential access or a direct access device.

backout
   A backward recovery (versus a forward recovery).

backup copy
   A duplicate of the primary image copy of one or more table spaces or partitions. A backup copy will be used in an attempted recovery of those spaces when an attempted recovery that uses the primary copy fails. A backup copy may be designated for a local site or for a recovery site.

base table
   A table that is created by the SQL CREATE TABLE statement and that holds persistent data, or a table that contains a LOB or XML column.
base table space
A table space that contains base tables.

bind
The process by which the output from the DB2 precompiler is converted to a usable control structure called a package or an application plan. During the process, access paths to the data are selected and some authorization checking is performed.

bootstrap data set
A VSAM data set that contains name and status information for DB2, as well as RBA range specifications, for all active and archive log data sets. It also contains passwords for the DB2 directory and catalog, and lists of conditional restart and checkpoint records.

BSAM
See basic sequential access method.

BSDS
See bootstrap data set.

buffer pool
Main storage reserved to satisfy the buffering requirements for one or more table spaces or indexes.

C
CAF
See call attachment facility.

call attachment facility
A DB2 attachment facility for application programs running in TSO or batch. The CAF is an alternative to the DSN command processor and provides greater control over the execution environment.

CART
Abbreviation for cartridge, a storage device that consists of magnetic tape, on supply and take-up reels, in a protective housing.

catalog
In DB2, a collection of tables that contains descriptions of objects such as tables, views, and indexes.

catalog table
Any table in the DB2 catalog.
change accumulation file
A change accumulation file accumulates, extracts, and stores log record data for a specified object or group of objects. The BMC R+/CHANGE ACCUM product produces change accumulation files.

CHECK
Check pending (a DB2 space status).

cHECK
A state of a table space or partition that prevents its use by some utilities and some SQL statements, because it can contain rows that violate referential constraints, table check constraints, or both.

checkpoint
A point at which DB2 records internal status information about the DB2 log that would be used in the recovery process if DB2 should abend.

cipher text
Data that has been converted to mask its meaning from an unauthorized recipient. Cipher text is the same as encrypted text.

clear text
Data in normal, readable form. COPY PLUS standard image copies are in clear text. Clear text is the same as plaintext.

clause
In SQL, a distinct part of a statement, such as a SELECT clause or a WHERE clause.

CLIST
Command list. A language for performing TSO tasks.

cлон table
A table that is structurally identical to a base table. The base and clone table each have separate underlying VSAM data sets, which are identified by their data set instance numbers.

clustering index
An index that determines how rows are physically ordered (clustered) in a table space. If a clustering index on a partitioned table is not a partitioning index, the rows are ordered in cluster sequence within each data partition instead of spanning partitions. Prior to DB2 Version 8, the partitioning index was required to be the clustering index.

cold start
A process by which DB2 restarts without processing any log records.

column
The vertical component of a table. A column has a name and a particular data type (for example, character, decimal, or integer).
command
A RECOVER PLUS command such as OPTIONS or REBUILD INDEX, DB2 operator command, or a DSN subcommand. Distinct from an SQL statement.

commit
The operation that ends a unit of work by releasing locks so that the database changes made by that unit of work can be perceived by other processes.

commit point
A point in time when data is considered consistent.

concurrency
The shared use of resources by multiple processes at the same time.

concurrent copy
A feature that allows a copy to be made of a data set while concurrent read and write activity is occurring against that data set.

conditional restart
A DB2 restart that is directed by a user-defined conditional restart control record (CRCR).

conditional restart control record
A control record that is used to direct a user-defined DB2 conditional restart.

coordinated recovery
A method of recovery in which backups, recoveries, and associated tasks are coordinated across multiple DBMSs and applications.

copy pending
A DB2 table space status.

coupling facility
An operating system hardware feature that facilitates communication between multiple DB2s in a data sharing group and the shared data storage.

CRCR
See conditional restart control record.

cross-system coupling facility
A facility that provides a communications link across systems.

D
DASD
See direct access storage device.
database
A collection of tables, or a collection of table spaces and index spaces.

database administrator
An individual responsible for the design, development, operation, safeguarding, maintenance, and use of a database.

database descriptor
An internal representation of a DB2 database definition which reflects the data definition found in the DB2 catalog. The objects defined in a database descriptor are table spaces, tables, indexes, index spaces, and relationships.

database management system
A software system that controls the creation, organization, and modification of a database and access to the data stored within it.

data control block
A control block that is used by access method routines in storing and retrieving data.

data definition name
In JCL, the name of the data definition (DD) statement that names a data set and provides all of the required information for that data set to be processed.

Data Facility Storage Management Subsystem
An operating environment that helps automate and centralize the management of storage.

data partition
A VSAM data set that is contained within a partitioned table space.

data-partitioned secondary index (DPSI)
A secondary index that is partitioned. The index is partitioned according to the underlying data.

data sharing
The ability of two or more DB2 subsystems to directly access and change a single set of data.

DBA
See database administrator.

DBADM
Database administration privileges.

DBCTRL
Database control privileges.

DBCS
See double-byte character set.
DBD
   See database descriptor.

DBID
   Database identifier.

DBMS
   See database management system.

DB2 catalog
   Tables maintained by DB2 that contain descriptions of DB2 objects such as tables, views, and indexes.

DB2 command
   An instruction to the DB2 subsystem that enables you to start or stop DB2, to display information about current users, to start or stop databases, to display information about databases, and so on.

DCB
   See data control block.

DD name
   See data definition name.

DD statement
   Data definition statement.

default value
   A predetermined value, attribute, or option that is assumed when no other is explicitly specified.

DFSMS
   See Data Facility Storage Management Subsystem.

DFSMS Concurrent Copy
   An IBM DFSMS feature that allows a copy to be made of a data set while concurrent read and write activity is occurring against that data set.

direct access storage device
   A device in which access time is independent of the location of the data.

directory
   The system database that contains internal objects such as database descriptors.
double-byte character set
A set of characters that is used by national languages such as Japanese and Chinese that has more symbols than can be represented by a single byte. Each character is two bytes in length, and therefore requires special hardware to be displayed or printed.

DSN1COPY
In DB2, a service aid that is used to copy table space data to a sequential data set or to copy a sequential data set to a table space.

E
EDITPROC
In DB2, an SQL edit procedure typically used for the compression or encryption of data.

ECKD device
Extended count-key-data device. A type of disk storage device.

encrypted text
Data that has been converted to mask its meaning from an unauthorized recipient. Encrypted text is the same as cipher text.

exception status
In DB2, an abnormal table space or partition status (for example, check pending, copy pending, or recover pending).

EXCP
Execute channel program.

exit routine
A user-written (or IBM-provided default) program that receives control from DB2 to perform specific functions. Exit routines run as extensions of DB2.

F
fallback
The process of returning to a previous image copy of a space after attempting unsuccessfully to use the most recent image copy of a space for recovery.

forward recovery
The process of reconstructing a data set from a particular point by restoring a saved version of the data set, then applying changes to it in the same order in which they were originally made.

free space
The total unused space in a page, that is, the space not used to store records or control information.
**full copy**

In DB2, a complete copy of a specified space or spaces. This can be an image copy or a DSN1COPY-type copy.

**G**

**GB**

Gigabyte (1,073,741,824 bytes).

**GDG**

See generation data group.

**generation data group**

A collection of data sets kept in chronological order. Each data set is a generation data set.

**group attachment name**

An alternative to subsystem ID for data sharing that allows the application to attach to any member in the group.

**group buffer pool**

DB2 buffer pool used for sharing access to pages between members of a data sharing group.

**I**

**ICF**

See integrated catalog facility.

**IDCAMS**

An IBM access method services program for data set and catalog structure maintenance.

**image copy**

In DB2, a replica of a physical object such as a table space or data set.

**incremental copy**

In DB2, a copy of only the changes that have been made to a specified space since the previous full or incremental copy was made.

**index**

A set of pointers that are logically ordered by the values of a key. Indexes can provide faster access to data and can enforce uniqueness on the rows in a table.

**index-controlled partitioning**

A type of partitioning in which partition boundaries for a partitioned table are controlled by values that are specified on the CREATE INDEX statement. Partition limits are saved in the LIMITKEY column of the SYSIBM.SYSINDEXPART catalog table.
index key
The set of columns in a table used to determine the order of index entries.

index partition
A VSAM data set that is contained within a partitioned index space.

index space
A page set used to store the entries of one index.

indoubt
A status of a unit of recovery. If DB2 fails after it has finished its phase 1 commit processing and before it has started phase 2, only the commit coordinator knows if this unit of recovery is to be committed or rolled back. At emergency restart, if DB2 does not have the information needed to make this decision, its unit of recovery is indoubt until DB2 obtains this information from the coordinator.

indoubt resolution
The process of resolving the status of an indoubt logical unit of work to either the committed or the rollback state.

inflight
A status of a unit of recovery. If DB2 fails before its unit of recovery completes phase 1 of the commit process, it merely backs out the updates of its unit of recovery when it is restarted. These units of recovery are termed inflight.

inline copy
A copy produced by a LOAD or REORG utility. The data set produced by the inline copy is logically equivalent to a full image copy produced by running a COPY utility with read-only access (SHRLEVEL REFERENCE).

installation SYSADM
The person having the highest level of authority within DB2. This authority includes SYSADM authority and the privilege of granting SYSADM authority to others. This authority is assigned at installation time.

Instant Snapshot
An Instant Snapshot is a non-standard, data set level copy that is made by using intelligent storage systems with COPY PLUS and XBM. Instant Snapshots may be used for recovery by RECOVER PLUS or RECOVERY MANAGER but by not DB2 RECOVER. Instant Snapshots are registered only in the BMCXCOPY table.

integrated catalog facility
A facility that provides for integrated catalog facility catalogs.

Interactive System Productivity Facility
An IBM licensed program that provides interactive dialog services.
**internal resource lock manager**
A subsystem that is used by DB2 to control communication and database locking.

**IRLM**
*See internal resource lock manager.*

**ISPF**
*See Interactive System Productivity Facility.*

**ISPF/PDF**
Interactive System Productivity Facility/Program Development Facility.

**J**

**JCL**
*See job control language.*

**JES**
Job Entry Subsystem.

**job control language**
A control language used to identify a job to an operating system and to describe the requirements of the job.

**JOBLIB**
*See job library.*

**job library**
A set of user-identified partitioned data sets used as the primary source of load modules for a job.

**K**

**K**
Kilobyte (1024 bytes).

**KB**
Kilobyte (1024 bytes).

**key**
A column or an ordered collection of columns identified in the description of a table, index, or referential constraint.

**key data set**
A data set that contains essential encryption key information and that is required for COPY PLUS encryption.
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

L

limit key
The highest value of the index key for a partition.

load module
A program unit that is suitable for loading into main storage for execution.

local
Refers to any object maintained by the local DB2 subsystem.

local site
A physical DB2 system installation where DB2 applications are installed and in use. Compare with recovery site.

lock
A means of controlling concurrent events or access to data. DB2 locking is performed by the IRLM.

lock duration
The interval over which a DB2 lock is held.

locking
The process by which the integrity of data is ensured. Locking prevents concurrent users from accessing inconsistent data.

log
A collection of records that describes the events that occur during DB2 execution and their sequence. The information recorded is used for recovery in the event of a failure during DB2 execution. In particular, the log records are used to reconstruct damaged tables and indexes.

logical page list
In DB2, a list of entries for pages that are logically in error. One of the types of exception statuses that RECOVERY MANAGER for DB2 is capable of recovering.

log record sequence number (LRSN)
A number DB2 generates and associates with each log record. DB2 also uses the LRSN for page versioning. The LRSNs generated by a given DB2 data sharing group form a strictly increasing sequence for each DB2 log and a strictly increasing sequence for each page across the DB2 group.

LPL
See logical page list.

LRSN
See Log Record Sequence Number.
M

Megabyte (1,048,576 bytes).

MERGECOPY
An IBM DB2 utility that is used to merge incremental image copies with previous incremental image copies or with a full image copy.

M
Megabyte (1,048,576 bytes).

modified-page indicator
A flag bit in a space map page that is set if the corresponding data page has been updated since the last copy was made of this table space.

N

nonpartitioned index
An index that is not physically partitioned. Both partitioning indexes and secondary indexes can be nonpartitioned.

nonpartitioned table space
In DB2, a simple table space (one that is not partitioned).

O

OBID
Identifier of the table space file descriptor.

P

package
An object containing a set of SQL statements that have been bound statically and that are available for processing. A package is sometimes also called an application package.

page
A unit of storage within a table space (4 KB, 8 KB, 16 KB, or 32 KB) or index space (4 KB). In a table space, a page contains one or more rows of a table.

page fixing
In virtual storage systems, marking a page as non-pageable so that it remains in real storage.

parse
In RECOVER PLUS, to analyze the options entered with the RECOVER PLUS commands and use the information to create a parameter list for the command processor.
partial recovery
Refers to a table space recovery to a prior point in time. The recovery can be to a prior copy, a prior quiesce point, or to a prior RBA.

partition
A portion of a page set. Each partition corresponds to a single, independently extendable data set. The maximum size of a partition depends on the number of partitions in the partitioned page set. All partitions of a given page set have the same maximum size.

partition-by-growth table space
Partition-by-growth table spaces are a type of universal table space (UTS). The usage of partition-by-growth table spaces is similar to a single table DB2 managed segmented table space. Partition-by-growth table spaces are managed by DB2. DB2 automatically adds a new data set when the database need more space to satisfy an insert. The table space begins as a single-partition table space and automatically grows additional partitions as needed to accommodate data growth. Partition-by-growth table spaces can grow up to 128 TB. The maximum size is determined by MAXPARTITIONS or DSSIZE value that you specified in DB2.

partitioned index
An index that is physically partitioned. Both partitioning indexes and secondary indexes can be partitioned.

partitioned table space
A table space subdivided into parts (based on index key range), each of which may be independently processed by utilities.

partitioning index
An index in which the leftmost columns are the partitioning columns of the table. The index can be partitioned or nonpartitioned.

phase
In RECOVER PLUS, a distinct part of the total process required for recovery. RECOVER PLUS phases are UTILINIT/ANALYZE, LOG INPUT, MERGE, RESTORE, UNLOAD, BUILD, SNAP, and UTILTERM.

plaintext
Data that is in normal, readable form. (COPY PLUS standard image copies are plaintext.) Plaintext is the same as clear text.

plan
See application plan.

point-in-time recovery
Refers to a table space recovery to a prior point in time. The recovery can be to a prior copy, a prior quiesce point, or to a prior RBA.
point of consistency
An RBA or LRSN to which a recovery can be made without jeopardizing the integrity of the data being recovered. Also called a quiesce point.

primary authorization ID
The authorization ID used to identify the application process to DB2.

primary copy
An image copy (either full or incremental) of one or more table spaces or partitions. This is the first copy to be used (when specified) in an attempted recovery of those table spaces. A primary copy may be designated for a local site or for a recovery site. Compare with backup copy.

PROC
In TSO, a command procedure.

PSID
Identifier of the table space page set descriptor.

Q
QSAM
Queued sequential access method.

quiesce point
See point of consistency.

R
RACF
See resource access control facility.

range-partitioned table spaces
Range-partitioned table spaces are a type of universal table space (UTS). Range-partitioned table spaces are based on partitioning ranges. The maximum size of a range-partitioned UTS is 128 TB. Range-partitioned UTS uses the segmented space map page organization. However, it contains a single table, which makes it similar to the regular partitioned table space.

RBA
See relative byte address.

reallocation
The process of respecifying the size of the space that is allocated to a data set.

rebind
To create a new application plan for an application program that has been bound previously. If, for example, you have added an index for a table accessed by your application, you must rebind the application to take advantage of that index.
recovery
   The process of rebuilding databases after a system failure.

recovery log
   See log.

recovery pending
   A status that prevents SQL access to a table space or index space that may need to be recovered.
   This status denotes that the table space has not yet been recovered to the current state.

recovery site
   A physical DB2 system installation that is designated as the backup system for a local site in the
   event that a disaster recovery is necessary.

RECP
   See recover pending.

referential integrity
   The condition that exists when all intended references from data in one column of a table to data
   in another column of the same or a different table are valid. Maintaining referential integrity
   requires enforcing referential constraints on all LOAD, RECOVER, INSERT, UPDATE, and
   DELETE operations.

register
   In DB2, to store information (critical to recovery) about an image copy in the SYSIBM.SYSCOPY
   table.

registration point
   The registration point is the log point (relative byte address (RBA) or log record sequence
   number (LRSN)) that is registered in the START_RBA column of SYSIBM.SYSCOPY or
   BMCXCOPY for an image copy. If a recovery is performed using the image copy, it is usually
   necessary to apply log records staring at the START_RBA.

relative byte address (RBA)
   A hex address (offset) that uniquely identifies a DB2 log record. The RBA is with respect to the
   start of the log (offset = zero).

remote
   Refers to any object maintained by a remote DB2 subsystem; that is, by a DB2 subsystem other
   than the local one. A remote view, for instance, is a view maintained by a remote DB2
   subsystem.

REORG pending
   A condition that restricts SQL access and most utility access to an object that must be
   reorganized.
REORP
  See REORG pending.

Resource Access Control Facility)
  For DB2, a security system that verifies authorization to access a specified data set.

RI
  See referential integrity.

RMGR
  RECOVERY MANAGER for DB2 product from BMC.

RO
  Read-only access. In DB2, a table space status that allows only read access to the table space.

rollback
  The process of restoring data changed by SQL statements to the state at its last commit point. All locks are freed.

RW
  Read-write access. In DB2, a table space status that allows both read and write access to the table space.

S

secondary authorization ID
  An authorization ID that is associated with a primary authorization ID by the authorization exit routine.

secondary index
  A nonpartitioning index on a partitioned table.

segmented table space
  A table space that is divided into equal-sized groups of pages called segments. Segments are assigned to tables so that rows of different tables are never stored in the same segment.

sequential data set
  A non-DB2 data set whose records are organized on the basis of their successive physical positions, such as on magnetic tape. Several of the DB2 database utilities require sequential data sets.

simple table space
  A table space that is neither partitioned nor segmented.
SMF
See system management facility.

SMS
See Storage Management Subsystem.

SQL
See Structured Query Language.

STARTDB
Start database privileges.

stacked tape
In backup and recovery, a tape on which successive image copies are stored.

storage group (stogroup)
A named set of DASD volumes on which DB2 data can be stored.

Storage Management Subsystem
An operating system component that is used to automate and centralize the management of storage by providing the storage administrator with control over data class, storage class, management class, storage group, and ACS routine definitions.

Structured Query Language
A database sublanguage used in querying, updating, and managing relational databases.

subsystem
A distinct instance of a RDBMS.

synchronization point (sync point)
A point in time when data is considered consistent and from which an application can restart if a failure occurs.

SYSADM
System administration privileges.

SYSCTRL
System control privileges.

SYSIBM.SYSCOPY
In DB2, a catalog table that stores (registers) information about each image copy made. The information stored is required for recovery purposes and includes the names of the subject database and spaces, the date and time of the copy, and the SHRLEVEL type.

SYSIBM.SYSLGRNX
In DB2, a table used to store log ranges associated with table space transactions.
system administrator
The person having the second highest level of authority within DB2. System administrators make decisions about how DB2 is to be used and implement those decisions by choosing system parameters. They monitor the system and change its characteristics to meet changing requirements and new data processing goals.

system management facility
An optional control program feature of OS/VS that provides the means for gathering and recording information that can be used to evaluate system usage.

system resources
In DB2, those items that are controlled by DB2, such as the BSDSs, the logs, and the catalog and directory.

table
A named data object consisting of a specific number of columns and some number of unordered rows. Synonymous with base table.

table check constraint
A user-defined constraint that specifies the values that specific columns of a base table can contain.

table-controlled partitioning
A type of partitioning in which partition boundaries for a partitioned table are controlled by values that are defined in the CREATE TABLE statement. Partition limits are saved in the LIMITKEY_INTERNAL column of the SYSIBM.SYSTABLEPART catalog table.

table space
A page set used to store the records in one or more tables.

thread
The DB2 structure that describes an application’s connection, traces its progress, processes resource functions, and delimits its accessibility to DB2 resources and services. Most DB2 functions execute under a thread structure.

time-sharing option
Provides interactive time sharing from remote terminals.

timestamp
A seven-part value that consists of a date and time expressed in years, months, days, hours, minutes, seconds, and microseconds.

TMM
Tape Mount Management.
**TSO**

*See* time-sharing option.

**U**

**universal table space (UTS)**

A table space that is both segmented and partitioned. A combination of partitioned and segmented table space schemes provides better space management as it relates to varying-length rows and improved mass delete performance. Universal table space types include range-partitioned and partition-by-growth table spaces.

**UT**

Utility-only access. In DB2, a table space status that allows only utilities to access the space.

**UTRO**

Utility/read-only. In DB2, a table space status that indicates a DB2 utility is executing against the table space but allows only read access by others to the table space.

**UTRW**

Utility/read-write. In DB2, a table space status that indicates a DB2 utility is executing against the table space but allows read and write access by others to the table space.

**UTUT**

Utility/utility. In DB2, a table space status that indicates a DB2 utility is executing against the table space and allows only other utilities to read and write to the table space.

**V**

**volser**

The name of a volume.

**VSAM**

Virtual storage access method.

**W**

**warm start**

The normal DB2 restart process which involves reading and processing log records so that data under the control of DB2 is consistent. Contrast with cold start.

**work file**

A temporary file that is used for the temporary storage of data that is being processed.

**X**

**XBM**

EXTENDED BUFFER MANAGER, a BMC product.
**XCF**

*See* cross-system coupling facility.
Index

Symbols

$C3DLOPT 535
&ATTACH symbolic variable 148, 180, 455
&DATE symbolic variable 148, 180, 455
&DAY symbolic variable 148, 180, 455
&DB symbolic variable 147, 180, 455
&DSNUM symbolic variable 147, 180, 455
&HOUR symbolic variable 148, 180, 455
&ICTYPE symbolic variable 148, 455
&INST symbolic variable 148, 181, 456
&JDATE symbolic variable 148, 180, 455
&JDAY symbolic variable 148, 180, 455
&JOBNAME symbolic variable 147, 180, 455
&LDSNUM symbolic variable 148, 180, 455
&LPART symbolic variable 148, 180, 455
&MINUTE symbolic variable 148, 180, 455
&MONTH symbolic variable 148, 180, 455
&PART symbolic variable 147, 180, 455
&SECOND symbolic variable 148, 180, 455
&SEQ symbolic variable 148, 181, 456
&SP symbolic variable 147, 180, 455
&SSID symbolic variable 148, 180, 455
&STEPNAME symbolic variable 147, 180, 455
&TIME symbolic variable 148, 180, 455
&TIME symbolic variable 147, 180, 455
&UID symbolic variable 148, 180, 455
&USERID symbolic variable 148, 180, 455
&UTIL symbolic variable 148, 180, 455
&V CAT symbolic variable 180
&YEAR symbolic variable 148, 180, 455
@x03DLOPT 535

Advanced Encryption Standard (AES) 462
advantages of LOGSORT strategy 419
advantages of sorting the log 418
AES (Advanced Encryption Standard) 462
AFRSOPTS installation options module 535
AFRDBG 279
AFRDUMP 278
AFRERR 278
AFR GDG data set 279, 454, 530
AFRMAIN execution module 291
AFROSUM 277
AFRPLAN 278
AFR PLAN data set 330
AFRPRINT 277
AFRSTMT 277
AFRSTMT data set 326
AFRSUMRY 277
AFRSUMRY data set 325
AFRTIME 278
AFRTRACE 278
allocating copy data sets 452
allocating output data sets 97, 146
allocation of EAV 151, 557
allocation of extended format sequential data set 151, 557
alphabetical listing of options 102, 594
ALTER for DB2 480, 482, 487, 491, 494, 500
AMSCAT option 541
ANALYZE ONLY option 325
ANALYZE option
ANALYZE ONLY 65
description 132, 189, 225, 235, 241
using to preview recovery resources and plans 65
ANSI Data Encryption Algorithm (DEA) 462
APF authorization 82, 535
archive log 411
ARCHIVE LOG MODE (QUIESCE) 508, 519, 530
archive quiesce points
  recovering from 198, 199
  recovering to 182
  scanning from 251
  scanning to 258
ASCII data support 433
audit controls 48

A

above-the-bar storage 79
access, shared 574
ACCUM command 383
ACF2 552
ACFORTSS installation option 552
ACHKP status 70
added partitions 460
adding comments 99
adding objects for restart 293
ADRDSSU utility 495
authority
- APF 82
- DATAACCESS 81
- DBADM 80
- DBCTRL 80
- DISPLAYDB 80, 81
- IMAGECOPY 81
- installation SYSADM 80
- RACF 81
- RECOVERDB 80
- STARTDB 80, 81
- STOPDB 80, 81
- SYSADM 80
- SYSCTRL 80
- system 81
- system DBADM 80
- to access DB2 subsystem 80
- to run RECOVER PLUS 80
- to start the database 80
- to stop the database 80

authorization
- mechanisms 552
- specifying CA ACF2 security or CA Top Secret security 552

automatic fallback 424

automatic sizing of output files 140
AUTOSIZE option 140, 551
AUX option 169, 175, 219, 250, 254
description 122
installation option description 555
auxiliary table spaces 70
AVGRECSZ option 58, 115

B

backing up BMC tables 567
BACKOUT AUTO option 61
BACKOUT AUTO option, description 125
BACKOUT option
- and indexes defined using an expression 429
- and LOBs 429
- and NOT LOGGED spaces 429
- and XML spaces 429
description 124, 259
example 364
for TOLOGPOINT 184, 259
for TORBA 184, 259
installation option description 556
recovery without image copies 59
restrictions 126, 164, 184, 201, 259
backout recovery 60
backup copies, making 199
benefits of RECOVER PLUS 46
bind qualifier 553
BINDQUALIFIER installation option 553
BMC Common DB2 repository tables 587

BMC Software, contacting 2
BMC utilities
- displaying status 568
- running concurrently 575
- terminating 569

BMCDICT table
- considerations 569
- contents 569
- maintaining 570

BMCHIST table
- and the HISTORY installation option 553
- backing up 567
- contents 570
- maintenance 572
- querying 568

BMCLGRNX table 572

BMCSORT
- DYNALLOC option, RETRY parameter 564
- installation options 561
- RETRY parameter of DYNALOC option 564
- retrying dynamic allocation of SORTWK 564

BMCSYNC table
- backing up 567
- clean up after job termination 297
- cleaning up RECOVER PLUS UNLOADKEYS 578
- concurrency 87
- considerations 577
- contents 573
- maintaining 577
- running utilities concurrently 575
- utilid deleted 273

BMCTRANS table 578

BMCUTIL table
- backing up 567
- clean up after job termination 297
- concurrency 87
- contents 580
- maintaining 582
- utilid deleted 273

BMCXCOPY table
- backing up 567
- contents 583
- maintaining 586
- querying 568
- registrations 68
- role in index backups 72
- build phase 311

BUILDINDEX option
description 238
- rebuilding nonpartitioned indexes 64
- building a nonpartitioned index 366
- building RECOVER PLUS jobs 268
CA ACF2 security 552
CA Top Secret security 552
Cabinet copies
   overview 74
   registration 74, 75
   strategies for use 444
CATALOG MANAGER for DB2 480, 482, 487, 491, 500
catalog, ICF 81
CATLG option 149
change accumulation file
   described 448
   dynamic sizing 140
   example 383
   grouping objects 448, 450
CHANGE MANAGER for DB2 480, 482, 487, 491, 494, 500, 501
changed limit keys 461
CHECKINT option
   command option description 117
   deciding values for 91
   effects on restart 295
   installation option description 540
   overriding installation value 91
checkpoint override parameter
   CHECKPT(NO) 274
   CHECKPT(PHASE) 274
   controlling overhead 273
   effects on restart 295
checkpoints, specifying the time between 117
CHECKPT option
   checkpoint override parameter 273
   effects on restart 295
   installation option description 540
   overriding installation value 91
CHECKPT(NO) 274
CHECKPT(PHASE) 274
cleaning up
   after a job termination 297
   after an UNLOADKEYS job 297
   BMCUTIL and BMCSYNC 297
   using the TERM parameter 273
CLONE option 189, 221, 232, 241, 255, 472
cloned objects
   CLONE option 472
   data sets 472
   limitations for RECOVER PLUS support 472
   recovering 472
commands, alphabetical listing 102, 594
comments coded in statements 99
compatibility
   BMC Software products 53
   DB2 resources 53
compressed indexes 474
COMPRESSED option 207
compressing data sets, when stacking to tape 153
compression, BMCDICT table 569
concurrent operation
   of UNLOADKEYS and BUILDINDEX 89
   recovering multiple partitions 411
   recovery of table space and indexes 417
   with IBM DB2 utilities 86
   with other BMC utilities for DB2 87
   with other RECOVER PLUS executions 89
consistent point, obtaining 518, 529
controlling output 277
conventions, documentation 23
    copies
    recovering from the last 198
    scanning from the last 251
copy data set ddnames 208, 209, 210
COPY PLUS
   long names in statements 100
   utility integration 54
copy registration, SHRLEVEL 528
copying cabinet copies 73
creating
   consistent database for query and review 501
   consistent set of copies at a point in time 529
   consistent set of copies at a point in time, OUTCOPY
      ONLY JCL 530
   copies from log and prior copies 527, 529
   DB2 objects 483, 487, 495, 502, 513
   image copy without reading a space 532
   migration images 519
   creating copies from recovery resources 69
   cumulative incremental image copies 90
   customer support 3

D

data compression, for tape data sets 153
data definitions, obtaining 482, 487, 491, 494, 501, 510, 520
Data Encryption Standard (DES) 462
data migration 497, 498
data set name option
   TOCOPY specification 186
   TOVOLUME CATALOG option 187
   TOVOLUME volumeSerialNumber option 187
data sets
   AFRDBG 279
   AFRDUMP 278
   AFRERR 278
   AFRGDG 279
   AFROSUM 277
   AFRPLN 278
   AFRPRNT 277
   AFRSTMT 277
   AFRSUMRY 277
   AFRTIME 278
   AFRTRC 278
   compression, for tape data sets 153
data sets (continued)
dynamic allocation 97, 146, 276
for log 282
for output copies 208, 209
for REBUILD INDEX 282
for RECOVER BUILDINDEX 285
for RECOVER INDEX 280
for RECOVER INDEXSPACE 280
for RECOVER TABLESPACE 280
for RECOVER UNLOADKEYS 284
for sorts 286
input image copies 280
interim for index keys 64
L001WKnn 286, 290
log sort work 282
LxxxWKnn 286, 290
optional 277
output image copies 280
reallocating 191
required 277
reusing 192, 227, 243
S001WKnn 284, 286
SKEYDDN option 284, 285
STOGROUP-defined 191, 226, 242
STOGROUP-defined, required authority 81
SxxxWKnn 283, 286
symbolic variables 180
SYSERR 278
SYSIN 277
SYSOUnnn 278, 283
SYSPICK 279
SYSPRINT 277
SYSSCAN 278
SYSSDUMP 278
tape stacking considerations 367
VCAT-defined 191, 226, 242
VSAM 192, 227
WORKDDN option 284
data sharing
considerations 457
group attach name 271
DATAACCESS authority 81
DATAACL installation option 559
DATAACL option 150
DATAMVR option 131, 552
date/time values
Daylight Savings Time adjustments 143, 548
TZRULE option 143, 548
Daylight Savings Time adjustments 548
DB2
authorizations needed for RECOVER PLUS 80
commands issued by RECOVER PLUS 84
group attach name in JCL 271
object status 274
RACF ID 81
subsystem ID (SSID) in JCL 271
subsystem identifier (SSID) 271
DB2 objects, creating 483, 487, 495, 502, 513
DB2 REPAIR VERSIONS 471
DB2 Version 8 consideration 269
DB2 versioning, steps to handle 469
DBADM authority 80
DBCTRL authority 80
DBID 485, 488, 489, 503, 512, 514, 521, 524
DD name construction
for nonpartitioned spaces 281
for partitioned spaces 281
DD statements
common to all RECOVER PLUS executions 277
for REBUILD INDEX 282
for RECOVER BUILDINDEX 285
for RECOVER INDEX 280
for RECOVER INDEXSPACE 280
for RECOVER TABLESPACE 280
for RECOVER UNLOADKEYS 284
in JCL 276
input image copies 280
L001WKnn 286, 290
log data sets 282
LxxxWKnn 286, 290
output image copies 280
S001WKnn 284, 286
SKEYDDN option 284, 285
specifying work data sets for index keys 423
STEPLIB 276
SxxxWKnn 283, 286
SYSUT1 284
tape mounts, stacked tapes 280
WORKDDN option 284
DDNAME 365
DEA (Data Encryption Algorithm 462
default option See installation options
DES (Data Encryption Standard 462
DFDSS utility 495
directing recovery output 369
DISPLAYDB authority 80, 81
displaying job status 297
displaying status of BMC utilities 568
DocID 473
documentation
conventions 23
related 22
summary of changes 26
DOPT See installation options
drives, maximum 545
drop recovery
dropped table from a simple table space 491
dropped table in segmented table space 486
overview 479
using pack dump backup 494
when image copies are available 482
DROPRECOVERY option
description 170
example of use 486, 489, 493, 495
restrictions 164
DSN1COPY with INCOPY 62
DSN1LOGP utility
eexample use 483
output example 492
using to find LRSN 491
DSN1PRNT utility 483, 488
DSNAME option 147, 204
DSNHDECFS for ssid 271
DSNUM option 169, 172, 250, 252
DSNUTILB load module 82
DSSIZE option 205
DST adjustments 548
dynamic allocation
by BMCSORT 131, 222, 233
of output data sets 97, 146
of temporary sort work data sets 131
option descriptions 147
when stacking to tape 456
dynamic allocation of output copies 359
dynamic grouping 476
dynamic sizing of output files 140

E

EARLYCAT option 112
EARLYRECALL option 112, 334
EATTR installation option 557
EATTR option 151, 557
EAV allocation 151, 557
elapsed time 46
eliminating interim data sets for index keys 64
ENCIPHER option 463
encrypted copies
Advanced Encryption Standard (AES) 462
ANSI Data Encryption Algorithm (DEA) 462
Data Encryption Standard (DES) 462
ENCIPHER option 463
key data set 463
key data set for 463, 552
KEYDSNAM installation option 463, 552
overview 462
registration 468
requirements 462
Triple Data Encryption Standard (TDES) 462
ENCRYPTED option 206
ERRCONT option 548
examples
building a nonpartitioned index 366
creating a consistent database for query and review 501
creating a consistent set of copies at a point in time 529
creating an image copy without reading a space 532
creating and dynamically allocating output copies 359
drop recovery using pack dump backup 494
drop recovery when image copies are available 482
dropped table in a segmented table space 487
extracting nonpartitioned keys 365
moving a table space to another subsystem 508, 518
moving an application to another subsystem 508
OUTCOPY ONLY 530, 532
overriding installation options 372
parallel index rebuilds 391, 397
QUIESCE 504
RECOVER PLUS jobs 319
recovering a dropped table 491
recovering a dropped table space 482
recovering to a non-DB2 data set 369
recovering to a specified copy 360
recovering to a specified LRSN 363
recovering to current state 334
recovering using a change accumulation file 383
recovery plan preview 325
simple JCL for RECOVER PLUS execution 268
using LOGSCAN command 387
using SIMULATE YES option 390
using tape-stacked data sets 367
using the MAINT parameter 384
using the MAXKSORT option 391, 397
EXEC statement
in RECOVER PLUS JCL 269
utility parameters 270
execution module (AFRMAIN) 291
execution phases
descriptions 311
for table space and index recovery 316, 317, 318
for table space recovery 312, 313, 315
using Instant Snapshots 318
with BACKOUT 317
execution plan 539
EXPDRT option 154
expiration date for tape copies 154
extended format sequential data set allocation 151, 557
extracted keys files work space requirements 290
extracting index keys from non-DB2 data set 231
extracting nonpartitioned keys 365

F

fallback recovery 424
features
OBID translation 48
OUTCOPY ONLY 50
running without SYSLOGRNX records 49
using non-registered input copies 49
features of RECOVER PLUS 46
fixed disk storage requirements 434
FlashCopy image copies 120, 196, 204, 432, 551
FLASHCOPY option 196
formats, syntax diagrams 24
forward recovery without image copy 62
FROMLOGPOINT option
description 198, 250, 253
LASTARCHQ 199, 251
LASTCOPY 198, 251
LASTQUIESCE 198, 251
LASTSHUTDOWN 199, 251
restrictions 164
syntax diagram 161, 612
X’logPoint’ 199, 252
FROMRBA option
description 198, 250, 253
LASTARCHQ 199, 251
LASTCOPY 198, 251
LASTQUIESCE 198, 251
LASTSHUTDOWN 199, 251
restrictions 164
syntax diagram 161, 612
X’logPoint’ 199, 252
GDG model 454
GDGs in copy data set names 454
group, group authorizations table 591
group attach name 271
GROUPAUTH table 591
groups 476
GRPOPTS table 590
GDG model 454
GDGs in copy data set names 454
group, group authorizations table 591
group attach name 271
GROUPAUTH table 591
groups 476
GRPOPTS table 590
Hardware compression, BMCDICT table 569
High-speed Structure Change
description 50
DSSIZE option description 205
MODEL option description 205
PIECESIZE option description 205
TRANSFORM option description 171
HISTORY option
and BMCHIST table 570
description of installation option 553
history relationship 123, 555
HPGOBID 485, 488
HPGRBRBA 484, 486
IBM Resource Access Control Facility. See RACF
ICF catalog 81
IDCAMS load module 82
identifying maintenance applied 272
IMAGCOPY authority 81
image copies, invalid 424
image copy
creating 69
recovering a partitioned table space 444
SHRLEVEL CHANGE copy 424
including comments 99
INCOPY option 481, 486, 499, 513, 515, 522, 525
description 195
object ID translation 48
restrictions 164
INCOPY specification
description 198, 204
syntax diagram 162, 613
INCR option 196
incremental copies of nonpartitioned indexes 73
incremental image copies, cumulative 90
INDEP INTABLESPACE option 369, 498
INDEP OUTSPACE option 369, 487, 489, 498, 505
INDEPENDENT INTABLESPACE option 219, 231
INDEPENDENT OUTSPACE option
description 178, 217, 239
syntax diagram 158, 609
INDEVT option 207
index backups, incremental copies of nonpartitioned indexes 73
index builds, multitasking 278, 283
index key sort concurrency 136, 439
index keys, extracting from non-DB2 data set 231
INDEX option 172, 174, 215, 252, 253
index rebuild 64
index sort work space requirements 287
index space status
actions for access to index space 85
initial 84, 85
required initial status 83
unacceptable values 85
index specification
multiple indexes 174, 175
multiple indexes for LOGSCAN 253
single index 171
single index for LOGSCAN 252
syntax diagram 157, 245, 608, 619
INDEXES option 168, 249
indexes, compressed 474
INDEXLOG option
description for OPTIONS command option 126
description of installation option 541
restart consideration with AUTO value 296
INDEXSPACE option 172, 175, 252, 254
inflight resolution recovery 398
initial status restoration 86
inline copies
performance 419
restrictions 55, 201
INLINE option 196
IBM Resource Access Control Facility. See RACF
ICF catalog 81
IDCAMS load module 82
identifying maintenance applied 272
IMAGCOPY authority 81
image copies, invalid 424
image copy
creating 69
recovering a partitioned table space 444
SHRLEVEL CHANGE copy 424
including comments 99
INCOPY option 481, 486, 499, 513, 515, 522, 525
description 195
object ID translation 48
restrictions 164
INCOPY specification
description 198, 204
syntax diagram 162, 613
INCR option 196
incremental copies of nonpartitioned indexes 73
incremental image copies, cumulative 90
INDEP INTABLESPACE option 369, 498
INDEP OUTSPACE option 369, 487, 489, 498, 505
INDEPENDENT INTABLESPACE option 219, 231
INDEPENDENT OUTSPACE option
description 178, 217, 239
syntax diagram 158, 609
INDEVT option 207
index backups, incremental copies of nonpartitioned indexes 73
index builds, multitasking 278, 283
index key sort concurrency 136, 439
index keys, extracting from non-DB2 data set 231
INDEX option 172, 174, 215, 252, 253
index rebuild 64
index sort work space requirements 287
index space status
actions for access to index space 85
initial 84, 85
required initial status 83
unacceptable values 85
index specification
multiple indexes 174, 175
multiple indexes for LOGSCAN 253
single index 171
single index for LOGSCAN 252
syntax diagram 157, 245, 608, 619
INDEXES option 168, 249
indexes, compressed 474
INDEXLOG option
description for OPTIONS command option 126
description of installation option 541
restart consideration with AUTO value 296
INDEXSPACE option 172, 175, 252, 254
inflight resolution recovery 398
initial status restoration 86
inline copies
performance 419
restrictions 55, 201
INLINE option 196
INLOG option
   description 197
   restrictions 164
INLOG specification, syntax diagram 160, 611
input copies, non-registered 49
input image copy data sets 280
INSEQNO option 207
installation 22
installation options
   $C30DOPT member 535
   ACFORTSS 552
   affecting restart 295
   alphabetical listing 537
   AMSCAT 541
   AUTOSIZE 551
   AUX 555
   BACKOUT 556
   BINDQUALIFIER 553
   CHECKIN 91, 295, 540
   CHECKPT 91, 295, 540
   DATACLASS 559
   DATAMVR 552
   EATTR 151, 557
   ERRCONT 548
   HISTORY 553
   INDEXLOG 541
   IXRECP 542
   KEYDSNAM 463, 552
   KSORTSHARE 554
   LOCCASEL 551
   LOCCPSEL 551
   MAXDRIVE 545
   MAXSORT 544
   MAXLOGS 545
   MAXSORT 553
   OPNDDB2ID 81, 539
   OUTCOPY 542
   PLANRECV 539
   PUBLICPLAN 553
   RCLTSK 540
   RDB2STAT 86, 540
   REMCPSEL 551
   RESINV 546
   SMCORE 545
   SORTDEVT 543
   SORTNUM 543
   STORCLASS 559
   TBUFFS 539
   TZRULE 548
   USEHDROBIDS 557
   WKUNIT 546
   WTOR 548
   XBMID 546
   ZIP 547
installation options, overriding 372
installation SYSADM authority 80
Installation System 535
Instant Snapshots
   general 62, 429
   migration 431
   OBIDXLAT 431
   registration 430
   restrictions 201, 431
   SNAP phase 430
   with other BMC Software utilities 432
invalid image copies 424
INVOLUME option 207
issuing a QUIESCE 504
IXEXPAND option 475
IXRECP option 128, 373, 542

J

JCL
   checkpoint override parameter 273
   DD statements 276
   EXEC statement 269
   group attach name 271
   input image copy data sets 280
   installation options module (AFROPTS) 275
   JOB statement 269
   JOBLIB statement 276
   message level parameter (MSGLEVEL) 273
   output image copy data sets 280
   RDB2STAT override parameter 274
   REGION parameter 269
   simple example for RECOVER PLUS execution 268
   STEPLIB DD statement 276
   subsystem ID (SSID) 271
   utility identifier (UTILID) 271
job restart 91
JOB statement
   in RECOVER PLUS JCL 269
   parameters 269
   JOBLIB statement 276
jobs
   building 268
   displaying status 297
   restarting 292
   starting 291
   terminating 273

K

key data set for encrypted copies 463, 552
contents 465
management 467
overview 463
requirements 463
key sorts, shared 139
key work data sets option restrictions 264
KEYDSNAM installation option 463, 552
keys sort work files 138, 283, 545, 554
KEYSORT option 131
keywords. See individual keywords
KSORTSHARE option
  description 139
  installation option description 554

L
L001WK DD statements 114
L001WKnn 286, 290
large object recovery 69
last copy
  recovering from 198
  scanning from 251
last shutdown
  recovering to 183
  scanning to 258
LASTARCHQ option 182, 199, 251, 258
LASTCOMMONQ option 182, 258
LASTCOPY option 186, 198, 251, 360
LASTQUIESCE option 182, 198, 251, 257, 363
LASTSHUTDOWN option 183, 199, 251, 258, 508, 519
limit keys, changed 461
limitations of RECOVER PLUS 55
LOAD LOG YES 197
LOAD LOG YES with REPLACE 49
LOB data, MEMLIMIT setting 79
LOBs
  and the BACKOUT option 124, 429
  exception status and LOBs 70
  general recovery 69
  point-in-time recovery 70
  recovery of LOG NO auxiliary spaces 70
local site copies, primary and backup 202
LOCALSITE option 188
LOCCASEL option 551
LOCCPSSEL option 551
log files, reading concurrently 451
log input phase 311
log marks, recovering to 143, 183
Log Master for DB2 483, 489, 491, 530
log points
  recovering from 199
  recovering to 181, 184
  scanning from 252
  scanning to 259
log range table 572
log sort space requirements 290
LOGAPPLY ONLY option 188, 260
logging attribute 124, 429
LOGONLY option 188, 260, 495
LOGOUmmn 286, 554
LOGPOINT LOGMARK logMarkName option 143
LOGPOINT X’logpoint’ option 143
logs, maximum 545
LOGSCAN command
  dependencies and prohibitions 247
  description 99
  example 387
  index space specification 252, 253
  overview 58
  report contents 58
  syntax diagram 244
  syntax diagrams 244, 245, 618, 619
SYSSCAN 58
  table space specification 248
LOGSCAN command options
  AUX 250, 254
  BACKOUT 259
  CLONE 255
  DSNUM 250, 252
  FROMLOGPOINT 250, 253
  FROMRBA 250, 253
  INDEX 252, 253
  INDEX (ALL) TABLESPACE 254
  INDEXSPACE 252, 254
  LOGAPPLY ONLY 260
  LOGONLY 260
  TOLOGPOINT 257
  TORBA 257
LOGSORT
  advantages 419
  strategy 418
LOGSORT option
  command option description 113, 188
  syntax diagram 159, 610
long names in SYSIN 100
LSRN, finding with DSN1LOGP utility 489
LxxWKnn 286, 290

M
MAINT parameter 384
MAINT restart parameter used in JCL 272
maintaining common utility tables 566
maintenance tracking 272
MAXDRIVE installation option 545
MAXDRIVES option
  description 120
  example 372
  use with MAXLOGS 451
MAXKSORT option
  description 136
  example 391, 397
  installation option description 544
  setting 439
MAXLOGS option
  command option description 121
  installation option description 545
  usage rules 451
  use with MAXDRIVES 451
MAXLSORT option
  description 138
  installation option description 553
  setting 441
MEMLIMIT parameter 79, 269
  recommended setting with LOB data 79
  recommended setting with XML data 79
  setting 79
MEMLIMIT system parameter 79
MERGE phase, parallel 138
merge phase 311
message level parameter (MSGLEVEL), in JCL 273
MGMTCLAS option 150
migrating data 501, 519
MODEL option
  data set symbolic variables 180, 240
  for extracting indexing keys 232
  for INDEPENDENT OUTSPACE specification 180
  for rebuilding indexes 218, 220, 240
MODELDCB option 149
moving an application to another subsystem 508
MSGLEVEL option 277
multiple commands, using 261
multiple data sets
  BACKOUT 184
  DSNUM ALL 173, 253
  forward recovery of index spaces 458
  INDEP OUTSPACE 179, 217, 239
  INDEXLOG AUTO 296
  Instant Snapshots 430, 431
  MAXKSORT 136, 544
  OUTCOPY 200
  REDEFINE 93, 191, 192, 226, 242
  using copies 444
multiple index specification 174, 175, 253
multiple RECOVER PLUS commands
  restrictions for nonpartitioned table spaces 264
  restrictions for partitioned table spaces 264, 265
multitasking index builds 278, 283
non-DB2 data sets
  for extracting index keys 232
  for rebuilding indexes 218, 220, 240
  for recovery 180
nonpartitioned index recovery
  restrictions 265
  single and multiple jobs 414
nonpartitioned index, building 366
nonpartitioned indexes, incremental copies 73
nonpartitioned keys, extracting 365
non-registered copy specification, syntax diagram 160, 611
non-registered input copies 49
NOSCRATCH option, for REDEFINE 191, 227, 243
NOSYSLGRNG option 116, 481
NOT LOGGED attribute 124, 429
NOWORKDDN option 221, 334
NOWORKDDN strategy 422
NUMREC option 115, 224, 234

O

OBID
  example use 488
  finding for dropped objects 483
  finding the value for 488
  for newly created objects 485
  representation in PGSOBD 485
  translation example 489
  using the default for the source 512, 514
  using the default for the target 521, 524
OBID translation, description 48
OBIDs, use of those in header 123, 557
OBIDXLAT option description
  restrictions 164
  syntax diagram 160, 611
  use 193, 481, 486, 487, 489, 493, 495, 498, 505, 516, 525
object identifiers, obtaining 483, 488, 503, 512, 514, 521
OBJECTSET option 168, 172, 174, 176, 216, 249, 252, 254, 255, 476
OBJECTSET specification
  description 176, 216, 255
  syntax rules 176, 216, 255
OBJSET_DEF table 589
OBJSET_SQL table 590
OBJSETS table 588
obtaining data definitions 482, 487, 491, 494
ON ERROR option, description 135
Online Consistent Copies
  and compressed indexes 474
  and FlashCopy 432
  and INCOPY 196
  and log apply 196
  and migration 497
  and TRANSFROM 50
  as additional resource 53
  cabinet copy support 74
Online Consistent Copies (continued)
DATAMVR option 76, 131
feature of Recovery Management solution 51
using 75
online schema evolution 460
online schema evolution support 460
operating environment 78
operating system requirements 78
OPNDB2ID installation option 81, 539
options
alphabetical listing 102, 594
ANALYZE 132, 189, 225, 235, 241
AUTOSIZE 140
AUX 122, 169, 175, 219, 250, 254, 555
AVGRECSZ 115
BACKOUT 124, 184, 259, 556
BACKOUT AUTO 125
BUILDINDEX 238
CATLG 149
CHECKINT 117
CLONE 189, 221, 232, 241, 255
COMPRESSED 207
DATACLAS 150, 559
DATAMVR 131
DROPRECOVERY 170
DSNAME 147, 204
DSNUM 169, 172, 250, 252
DSSIZE 205
EARLYCAT 112
EARLYRECALL 112
EATTR 151, 557
ENCIPHER 463
ENCrypted 206
EXPDT 154
FLASHCOPY 196
FROMLOGPOINT 198, 250, 253
FROMRBA 198, 250, 253
HISTORY 553
INCOPY 195, 198, 204
INCR 196
INDEPENDENT INTABLESPACE 219, 231
INDEPENDENT OUTSPACE 178, 217, 239
INDEVT 207
INDEX 172, 174, 215, 252, 253
index space specification 171, 174, 175, 252, 253
INDEXES 168, 249
INDEXLOG 126
INDEXSPACE 172, 175, 252, 254
INLINE 196
INLOG LOGPOINT 197
INLOG RBA 197
INSEQNO 207
INVOLUME 207
IXRECP 128
KEYSORT 131
KSORTSHARE 139, 554
LOCALSITE 188
LOGAPPLY ONLY 188, 260
LOGONLY 188, 260
LOGPOINT LOGMARK logMarkName 143
LOGPOINT X’logpoint’ 143
LOGSORT 113, 188
MAXDRIVES 120
MAXKSORT 136, 439
MAXLOGS 121
MAXLSORT 138, 441, 553
MGMTCLAS 150
MODELDCB 149
NOCOPYPEND 197, 227
NOEARLYCAT 112
NOEARLYRECALL 112
NOSYSLOGNG 116
NOWORKDDN 221
NUMREC 115, 224, 234
OBIDXLAT 193
OBJECTSET 168, 172, 174, 176, 216, 249, 252, 254, 255
ON ERROR 135
OUTCOPY 121, 199
OUTCOPYDDN 208
OUTCOPYDSN 208, 209
PART 220, 230
PIECESIZE 205
RECOVERYDDN 209
RECOVERYDSN 209
RECOVERYPOINT 140
RECOVERYSITE 189
REDEFINE 191, 226, 242
REGISTER 202
RESETRTS 203
RESINV 119
RESOURCE SELECTION 120
RETPD 153
REUSE 192, 227, 243
SIMULATE YES 66, 134
SKEYDDN 232, 241, 284, 285
SMCORE 119
SNAPSHOT 196
SORTDEVT 114, 131, 223, 233
SORTDIAG 118
SORTDYN 119
SORTNUM 114, 132, 223, 233
SPACE 151, 204
STACK 152
STOGROUP 117
STORCLAS 150, 559
table space specification 168, 219, 230, 240, 248
TABLESPACE 168, 219, 230, 240, 248
TIMESTAMP 143
TIMESTAMP with ENCRYPTED 206
TOCOPY 185
TOCOPY specification 185
TOLOGPOINT 181, 257
TORBA 181, 257
TRANSFORM 50, 171
options (continued)
TRTCH 129, 153
UNIT 148
UNITCNT 150
UNLOADKEYS 229
UPDATE VERSIONS 192
URIDDDN 142
USEACCUM 116
USEHDROBIDS 123
USEORDER 118
VOLCNT 149
VOLUMES 151
WORKDDN 221
WTOR 135
XBMID 129
ZIIP 130
OPTIONS command
description 96, 109, 372
syntax diagrams 109, 110, 111, 601, 602, 603
OPTIONS command options
ANALYZE 132
AUTOSIZE 140
AUX 122
AVGRECSZ 115
BACKOUT 124
BACKOUT AUTO 125
CHECKINT 117
DATAMVR 131
EARLYCAT 112
EARLYRECALL 112
INDEXLOG 126
IXRECP 128
KEYSORT 131
KSORTSHARE 139
LOGPOINT LOGMARK logMarkName 143
LOGPOINT ‘x’logpoint’ 143
LOGSORT 113
MAXDRIVES 120
MAXKSORT 136, 439
MAXLOGS 121
MAXLSORT 138, 441
NEARLYCAT 112
NEARLYRECALL 112
NOSYSLGRLNG 116
NUMREC 115
ON ERROR 135
OUTCOPY 121
RECOVERYPOINT 140
RESINV 119
RESOURCE SELECTION 120
SIMULATE YES 66, 134
SMCORE 119
SORTDEVT 114, 131
SORTDIAG 118
SORTDYN 119
SORTNUM 114, 132
STOGROUP 117
TIMESTAMP 143
TRTCH 129
URIDDDN 142
USEACCUM 116
USEHDROBIDS 123
USEORDER 118
WTOR 135
XBMID 129
ZIIP 130
options, installation. See installation options
OS requirements 78
OUTCOPY ONLY
description 50
restrictions 201
OUTCOPY option
description 121, 199
for RECOVER TABLESPACE 199
installation option description 542
use 359, 486, 496, 509, 516, 519, 525, 530, 532
OUTCOPY specification syntax diagram 161, 612
OUTCOPYDDN option
description 208
example 359
syntax diagram 162, 613
OUTCOPYDSN option, description 208
OUTPUT command
description 97
example 359
option descriptions 145
restrictions on use 146
syntax diagrams 145, 146, 604
OUTPUT command options
CATLG 149
DATACLAS 150, 559
DSNAME 147
ENCIPHER 463
EXPDT 154
for disk and tape 148
for disk data sets only 151
for tape data sets only 152
MGMTCLAS 150
MODELDCB 149
name 147
RETPD 153
SPACE 151
STACK 152
STORCLAS 150
VOLCNT 149
VOLUMES 151
output data set
dynamic allocation 97
names in OUTPUT statement 147
output descriptor
- data set name 147
- default volume list 151
- descriptor name 147
- expiration date 154
- model DCB 149
- number of volumes 149
- operating system catalog directive 149
- options for disk and tape 148
- options for disk data sets only 151
- options for tape data sets only 152
- output allocation units 151, 204
- output device name 148
- retention period 153
- SMS data class name 150, 559
- SMS management class name 150
- SMS storage class name 150, 559
- stacking copies to tape 152
- output files, dynamic sizing 140
- output image copy data sets 280
- output, controlling 277
- overhead, controlling checkpoints 273
- overriding installation option values 372
- overview
  - audit controls 48
  - of RECOVER PLUS 44
  - operations 51
  - reduced elapsed time 46
  - testing recovery procedures 50

parallel index key sort 136, 439
parallel index key sort, diagram 313, 315
parallel index rebuilds 312, 314, 391, 397
parallel index sorts 312, 314
parallel MERGE 441
parallel MERGE phases 138
parameter used in JCL
  - checkpoint override parameter 273
  - DB2 group attach name (SSID) 271
  - DB2 subsystem identifier (SSID) 271
  - installation options module (AFROPTS) 275
  - message level (MSGLEVEL) 273
  - RDB2STAT 274
  - restart parameter 271
  - utility identifier (UTILID) 271
parameters
  - MEMLIMIT 79
  - system 79
PART option 220, 230
partitioned index recovery
  - in multiple jobs 413
  - restrictions 264
partitioned table space considerations 411

partitions
  - added 460
  - rotated 461
performance considerations 409
PGSObd 485
phases
  - descriptions 311
  - of RECOVER PLUS 292
  - restarts allowed 292
SNAP 430
PIECESIZE option 205
PIT_RBA 421
PLANRECV installation option 539
point of consistency, obtaining 508, 518, 529
point-in-time recovery specification
  - description 181, 257
  - strategies 426
  - syntax diagram 159, 610
previewing recovery activity
  - discussion 65
  - example 325
  - option description 189, 235, 241
primary copies, local and recovery site 202
processing, fallback 424
PRODREG table 591
product support 3
PSID 485, 488, 489, 503, 512, 514, 521, 524
PSRCP status 83
PUBLIC privilege 553
PUBLICPLAN installation option 553

Q
QUIESCE 504, 508, 519, 529
quiesce points
  - recovering from 198
  - recovering to 182
  - scanning from 251
  - scanning to 257, 258

R
R+/CHANGE ACCUM
  - change accumulation file, using 383, 448
  - utility integration 54
RACF authority 81
RACF ID 81
RBA, finding with DSN1LOGP utility 489
RBDP index 373
RCLTSK option 540
RDB2STAT option
  - general 86
  - installation option description 540
RDB2STAT override parameter
in JCL 274
RDB2STAT(NO) 275
RDB2STAT(RO) 275
RDB2STAT(RW) 275
RDB2STAT(YES) 275
RDB2STAT(NO) override parameter 275
RDB2STAT(RO) override parameter 275
RDB2STAT(RW) override parameter 275
RDB2STAT(YES) override parameter 275
reallocation
data sets 191
STOGROUP-defined data sets 191, 226, 242
VCAT-defined data sets 191
VSAM data sets 192, 227
real-time statistics (RTS), resetting for OUTCOPY 203
REBUILD INDEX command
description 98
option descriptions 212
restrictions on use 261
syntax diagram 213, 615
use 518
REBUILD INDEX command options
ANALYZE 225
AUX 219
CLONE 221
INDEPENDENT INTABLESPACE 219
INDEPENDENT OUTSPACE 217
INDEX 215
NOCOPYPEND 227
NOWORKDDN 221
NUMREC 224
PART 220
REDEFINE 226
REUSE 227
SORTDEVT 223
SORTNUM 223
TABLESPACE 219
WORKDDN 221
rebuilding indexes
all associated with a table space 215
by partition 215, 220
RECOVER 593
RECOVER BUILDINDEX command
description 99
example 366
option descriptions 237
restrictions on use 262
syntax diagram 237, 617
RECOVER BUILDINDEX command options
ANALYZE 241
BUILDINDEX 238
CLONE 241
INDEPENDENT OUTSPACE 239
REDEFINE 242
REUSE 243
SKEYDDN 241, 285
TABLESPACE 240
RECOVER INDEX command
AUX option 175
dependencies and prohibitions 164, 165
description 97
index space specification 171, 174
restrictions on use 164
syntax diagrams 155, 157, 158, 606, 608, 609
use 506, 508
RECOVER INDEX command options
ANALYZE 189
AVGRECSZ 115
CLONE 189
COMPRESSED 207
DSNAME 204
DSNUM 172
FROMLOGPOINT 198
FROMRBA 198
INCOPY 195, 204
INCR 196
INDEPENDENT OUTSPACE 178
INDEVT 207
INLINE 196
INSEQNO 207
INVOLUME 207
LOCALSITE 188
LOGAPPLY ONLY 188
LOGONLY 188
LOGSORT 188
NOCOPYPEND 197
OBIDXLAT 193, 195
OBJECTSET 172, 174, 252, 254
OUTCOPY 199
OUTCOPYDDN 208
OUTCOPYDSN 208
RECOVERYDDN 209
RECOVERYDSN 209
RECOVERYSITE 189
REDEFINE 191
REGISTER 202
REUSE 192
SNAPSHOT 196
TOCOPY 185
TOLOGPOINT 181
TORBA 181
RECOVER INDEXSPACE command
dependencies and prohibitions 164, 165
description 97
index space specification 171, 175
restrictions on use 164
syntax diagrams 155, 157, 158, 606, 608, 609
RECOVER INDEXSPACE command options
ANALYZE 189
AVGRECSZ 115
CLONE 189
RECOVER INDEXXSPACE command options (continued)
DSNAME 204
DSNUM 172
DSSIZE 205
FLASHCOPY 196
FROMLOGPOINT 198
FROMRBA 198
INCOPY 195, 204
INCR 196
INDEPENDENT OUTSPACE 178
INDEVT 207
INLINE 196
INSEQNO 207
INVOLUME 207
LOCALSITE 188
LOGAPPLY ONLY 188
LOGONLY 188
LOGSORT 188
NOCOPYPEND 197
OBIDXLAT 193, 195
OUTCOPY 199
OUTCOPYDDN 208
OUTCOPYDSN 208
PIECESIZE 205
RECOVERYDDN 209
RECOVERYDSN 209
RECOVERYSITE 189
REDEFINE 191
REGISTER 202
REUSE 192
SNAPSHOT 196
TOCOPY 185
TOLOGPOINT 181
TORBA 181
TRANSFORM 171

RECOVER OBJECTSET command options
OBJECTSET 176, 216, 255

RECOVER PLUS commands
general restrictions for using multiple commands 261
restrictions for STOGROUP-defined spaces 263
using multiple commands in a single job 261

RECOVER TABLESPACE command
dependencies and prohibitions 164
description 97
restrictions on use 164
syntax diagrams 155, 156, 160, 606, 607, 611
table space specification 168

RECOVER TABLESPACE command options
ANALYZE 189
AUX 169
AVGRECSZ 115
CLONE 189
DROPRECOVERY 170
DSNAME 204
DSNUM 169
DSSIZE 205
ENCRYPTED 206
FLASHCOPY 196
FROMLOGPOINT 198
FROMRBA 198
INCOPY 195, 198, 204
INCR 196
INDEPENDENT OUTSPACE 178
INDEVT 207
INDEXES 168, 249
INLINE 196
INLOG LOGPOINT 197
INLOG RBA 197
INSEQNO 207
INDEVT 207
LOCALSITE 188
LOGAPPLY ONLY 188
LOGONLY 188
LOGSORT 188
NOCOPYPEND 197
OBIDXLAT 193
OBJECTSET 168, 249
OUTCOPY 199
OUTCOPYDDN 208
OUTCOPYDSN 208
PIECESIZE 205
RECOVERYDDN 209
RECOVERYDSN 209
RECOVERYSITE 189
REDEFINE 191
REGISTER 202
RESETRTS 203
REUSE 192
SNAPSHOT 196
SPACE 204
TIMESTAMP with ENCRYPTED 206
TOCOPY 185
TOLOGPOINT 181
TORBA 181
TRANSFORM 171
UPDATE VERSIONS 192

RECOVER UNLOADKEYS command
description 98
option descriptions 228
syntax diagram 228, 616

RECOVER UNLOADKEYS command options
ANALYZE 235
CLONE 232
INDEPENDENT INTABLESPACE 231
NUMREC 234
PART 230
SKEYDDN 232, 284, 285
SORTDEVT 233
SORTNUM 233
TABLESPACE 230
UNLOADKEYS 229

RECOVERDB authority 80
recovery
   after added partitions 460
   after limit key changes 461
   change accumulation file, using 448
   from a log point 199
   from the last archive quiesce point 199
   from the last copy 198
   from the last image copy 198
   from the last quiesce point 198
   from the last shutdown 199
   LOBs 69, 429
   multiple indexes 437
   multiple partitions concurrently 411
   nonpartitioned indexes 265
   output 369
   partitioned indexes 264
   plan 325
   rotated partitions 461
   simulation 390
   to a log mark 143, 183
   to a log point 181, 184
   to archive quiesce points 182
   to last shutdown 183
   to quiesce points 182
   using a full volume backup 90
   using BUILDINDEX 64
   using incremental image copies 90
   using NOWORKDDN 422
   using only the log 62
   using SHRLEVEL CHANGE copies 424
   using UNLOADKEYS using 64
   using UNLOADKEYS/BUILDINDEX 423
   while maintaining activity while 68
   without an image copy 62
Recovery Management for DB2 44, 45, 50, 140
RECOVERY MANAGER for DB2 530
RECOVERY MANAGER groups 476
RECOVERY MANAGER utility integration 54
recovery plan, option descriptions 132, 189, 225, 235, 241
recovery simulation 66, 134, 167, 214, 229, 238
recovery site copies, primary and backup 202
recovery strategies
   building nonpartitioned indexes on partition table space 64
   compressed indexes 474
   eliminating interim data set for index keys 64
   fallback 424
   large partitioned objects 450
   large table spaces 450
   LOGSORT 418
   maintaining activity 68
   planning recovery resources 449
   previewing recovery activity 65
   R+/CHANGE ACCUM, using 448
   recovery with an image copy 62
   recovery with an Instant Snapshot 62
   resource selection 64
   separate change accumulation files, using 450
   separate copies, using 450
   sorting the log 418
   testing recovery 65
   using BUILDINDEX 64
   using UNLOADKEYS 64
   recovery, local
      BACKOUT AUTO option 61
      backout recovery 60
   RECOVERYDDN option
      description 209
      syntax diagram 163, 614
   RECOVERYDSN option, description 209
   RECOVERYPOINT 71
   RECOVERYPOINT option, description 140
   RECOVERYSITE option 189
   RECP status 373
   REDEFINE option 191, 226, 242
   redirection of output 369
   reduced elapsed time 46
   REGION parameter 269
   REGISTER option 202
   registering index backups 72
   registration in the BMCXCOPY table 68
   related publications 22
   REMCPSEL option 551
   removing objects for restart 293
   REORG LOG YES 49, 197
   REPAIR VERSIONS 471
   repository tables 587
   requesting jobs, restarting 91
   requirements
      extracted keys files work space 290
      fixed disk storage 287
      index sort work space 287
      log sort space 290
      sort work space 287, 290
      virtual memory 287
   RESETRTS option 203
   residing in APF library 82
   RESINV option 119, 546
   RESOURCE SELECTION option 120
   resource selection, using 64
   restart
      affected by installation options 91
      at a MERGE phase 292
      at beginning of LOG INPUT phase 292
      at the beginning of the last incomplete phase 273
      considerations when creating output copies 294
restart (continued)
during BUILD phase 293
from last incomplete phase 273, 292
from last sync point 273
from point of failure 294
installation options that affect 295
limitations 222
LOGSORT restriction 418
RECOVER PLUS job 292
removing/adding objects 293
syntax options that affect 293
table for RECOVER PLUS phases 292
transform 295
using checkpoints for restart 91
using INDEXLOG AUTO 296
using sync restart 272
utility parameters that affect 271

restart parameter
MAINT 272
NEW 272
NEW/RESTART 272
NEW/RESTART(PHASE) 272
RESTART 273
RESTART(PHASE) 273
TERM 273

restoration of initial status 86, 275
restrictions
for key work data sets 264
for multiple RECOVER commands 261, 262, 264
for nonpartitioned table spaces 264
for partitioned table spaces 264, 265
for REBUILD INDEX 261
for RECOVER BUILDINDEX 262
for RECOVER PLUS 55
for SKEYDDN 264
for SORTDEVT 264
for SORTNUM 264
for STOGROUP-defined spaces 263
for WORKDDN 264
using LOGSORT 419
using multiple commands 261
retention period, tape copies 153
RETDP option 153
REUSE option 192, 227, 243
reusing data sets 192, 227, 243
revision bars 26
rotated partitions 461
RTS, resetting for OUTCOPY 203
running BMC utilities concurrently 575

S
S001WKnn 284, 286
scan
from a log point 252
from the last archive quiesce point 251
from the last copy 251
from the last image copy 251
from the last quiesce point 251
from the last shutdown 251
to a log point 257, 259
to archive quiesce points 258
to last shutdown 258
to quiesce points 257, 258

secondary extents, sliding-scale calculation 178, 217, 239
security
specifying 552
specifying CA ACF2 security or CA Top Secret security 552
setting SMCORE installation option values 437
shared key sorts 139
SHRLEVEL CHANGE copies used for recovery 424
SHRLEVEL column, in BMCSYNC table 574
SHRLEVEL of copies produced 528
shutdown
recovering from 199
recovering to last 183
scanning from 251
scanning to last 258
SIMRBLD 214
SIMRBLD command 390
SIMRCVR 167, 214, 229, 238
SIMRCVR command 390
SIMULATE option 66
SIMULATE YES option, description 66, 134
SIMULATE YES option, example use 390
simulating recovery 390
simulation 66
simulation mode 66, 134, 167, 214, 229, 238
SKEY DD statement 365
SKEYDDN option
description 232, 241
restart requirements 293
restrictions 264
unloaded keys file space requirements 291
sliding-scale calculation, for secondary extents 178, 217, 239
SMCORE installation option 437, 545
SMCORE option on the OPTIONS statement 119
SNAP phase 430
snap phase 311
SNAPSHOT option 196
sort concurrency 138, 441
sort message files 137, 138, 283, 286, 544, 554
sort options 118, 119
sort performance 433
sort sizes, determining 433
sort tasks
management 312
parallel 312, 314
sort work data set
  defining 286
  options for REBUILD INDEX 222
  options for RECOVER UNLOADKEYS 233
sort work space
  allocating 443
  requirements 287, 290, 435
SORTDEV option
  description 114, 131, 223, 233
  example 372
  general information 263
  installation option description 543
  restrictions 264
SORTDIAG option 118
SORTDYN option 119
sorting the log 418
SORTNUM option
  description 114, 132, 223, 233
  example 372
  general information 263
  installation option description 543
  restrictions 264
SPACE option 151, 204
specification
  EXEC statement 269
  JOB statement 269
SPUFI 485
SQL statements
  deleting rows from the BMCDICT table 570
  deleting rows from the BMCHIST table 572
  deleting rows from the BMCSYNC table 578, 582
  deleting rows from the BMCSYNC table for
    RECOVER UNLOADKEYS 578
  deleting rows from the BMCCOPY table 586
  displaying BMC utilities 568
  querying BMCHIST table 568
  querying BMCCOPY table 568
  terminating BMC utilities 569
STACK option 152
stacked tape
  change accumulation file, using 450
  image copies, dynamic allocations 456
  mounts 280
  ordering copies 445
  strategies in using copies 445
START_RBA 421
STARTDB authority 80, 81
starting a new RECOVER PLUS execution 272
starting a RECOVER PLUS job 291
starting a space after recovery 86
status
  displaying for jobs 297
  index space-unacceptable 85
  table space-unacceptable 84
  when RECOVER PLUS job is terminated 297
status of BMC utilities 568
STEPLIB DD statement 276
STOGROUP option 117
STOGROUP-defined data sets
  for index spaces 81
  reallocating 191, 226
STOPDB authority 80, 81
storage
  fixed disk requirements 287, 434
  virtual memory requirements 434
  virtual, specified by REGION parameter 269
storage, virtual 79
STORCLAS installation option 559
STORCLAS option 150
strategy
  LOGSORT 418
  NOWORKDDN 422
  UNLOADKEYS/BUILDINDEX 423
  subsystem ID (SSID) in JCL 271
  summary of product changes 26
  support, customer 3
SxxxWKnn 138, 283, 286, 545, 554
symbolic variables
  in copy data set names 455
  with INDEP OUTSPACE option 180, 218, 240
  with OUTPUT command 147
  with RECOVER INDEX 180
  with RECOVER INDEXSPACE 180
  with RECOVER TABLESPACE 180
syntax diagrams
  FROMLOGPOINT specification 161, 612
  FROMRBA specification 161, 612
  INCOPY specification 162, 613
  INDEPENDENT OUTSPACE specification 158, 609
  index specification 157, 158, 245, 608, 609, 619
  LOGSCAN command 244, 618
  LOGSORT specification 111, 159, 603, 610
  non-registered copy or INLOG 160, 611
  OBIDXLAT specification 160, 611
  OPTIONS command 109, 110, 601, 602
  OUTCOPY specification 161, 612
  OUTCOPYDDN specification 162, 613
  OUTPUT command 145, 146, 604
  point-in-time recovery specification 159, 610
  REBUILD INDEX command 213, 615
  RECOVER BUILDINDEX command 237, 617
  RECOVER INDEX command 155, 606
  RECOVER INDEXSPACE command 155, 606
  RECOVER TABLESPACE command 155, 606
  RECOVER UNLOADKEYS command 228, 616
  RECOVERYDDN specification 163, 614
  SORT specification 111, 603
  STOGROUP specification 111, 603
  table space specification 156, 245, 607, 618
  TOCOPY specification 159, 610
  TOLOGPOINT option 159, 610
  TORBA option 159, 610
  syntax, format for diagrams 24
  SYSADM authority 80
tape copies
  expiration date 154
  retention period 153
Tape Mount Management 153
tape mounts, stacked 280
tape pick list 65
tape stacking example 367
TBUFFS option 539
TDES (Triple Data Encryption Standard) 462
technical support 3
temporal tables 123, 555
TERM restart parameter 273
terminating BMC utilities 569
termination, RECOVER PLUS job 297
termination, RECOVER PLUS job 273
testing recovery strategy 50, 66
timestamp recovery 140, 398
TMM (Tape Mount Management) 153
TOCOPY option
description 185
example 360
syntax diagram 159, 610
TOCOPY specification
data set name option 186, 187
description 185
LASTCOPY option 186
restrictions 164
TOSEQNO option 187
TOVOLUME CATALOG option 187
TOVOLUME volumesSerialNumber option 187
TOVOLUME volumesSerialVolume option 187
TOLOGPOINT LOGMARK logMarkName 183
TOLOGPOINT option 363, 483, 489, 505
BACKOUT 184, 259
description 181, 257
LASTARCHQ 182, 258
LASTCOMMONQ 182, 258
LASTQUIESC 182, 257
LASTSHUTDOWN 183, 258
X’logPoint’ 184, 259
Top Secret security 552
TORBA 508
TORBA LOGMARK logMarkName 183
TORBA option 363, 505, 508, 519
BACKOUT 184, 259
description 181, 257
LASTARCHQ 182, 258
LASTCOMMONQ 182, 258
LASTQUIESC 182, 257
LASTSHUTDOWN 183, 258
X’logPoint’ 184, 259
TOSEQNO option for TOVOLUME volumeSerialNumber
TOVOLUME CATALOG option 187
TOVOLUME option 187
TOVOLUME volumesSerialNumber option 187
tracking maintenance applied 272
Transformer, restarting 295
Triple Data Encryption Standard (TDES) 462
TRTCH option 129, 153
TZRULE option 548

U
UNIT option 148
UNITCNT option 150
unload phase 311
unloading indexes by partition 230
UNLOADKEYS entries, cleaning up 578
UNLOADKEYS option
and indexes defined using an expression 56, 229
description 229
UNLOADKEYS/BUILDINDEX strategy 423
UPDATE VERSIONS option 192
URIDDDN option 142
USEACCUM option 116, 383
USEHDROBIDS installation option 557
USEHDROBIDS option 123
USEORDER option 118
utility database 565
utility identifier (UTILID) in JCL 271
utility integration
COPY PLUS 54
DB2 resources 53
overview 53
R+/CHANGE ACCUM 54
RECOVERY MANAGER 54
utility parameter
checkpoint override parameter 273
group attach name 271
installation options module (AFROPTS) 275
message level 273
on the EXEC statement 270
RDB2STAT 86
RDB2STAT override parameter 274
restart parameter 271
subsystem ID (SSID) 271
utility identifier (UTILID) 271

V
VCAT-defined data sets
option ignored (REDEFINE) 226, 242
reallocating 191
version information 469
versioning relationship 123, 555
versioning, steps to handle 469
virtual memory requirements 287, 434
virtual storage 79
virtual storage used by RECOVER PLUS 269
VOL=SER parameter 295
VCAT-defined data sets

VOLCNT option 149
VOLUMES option 151
VSAM data sets
and REDEFINE NO 227
reallocating 192

W
WKUNIT installation option 546
work data set ddname 221
work data sets, defining for sorts 286
work file 365
work space requirements 290
WORKDDN option
description 221
restrictions 264
working with ASCII data 433
WTOR installation option 548
WTOR option on the OPTIONS command 135

X
XBM subsystem ID 129, 546
XBMID option 129, 546
XML data, MEMLIMIT setting 79
XML objects
DocID 473
NodeID 473
recovering 473

Z
zIIP enablement 547
ZIIP installation option 547
ZIIP option
and the OPTIONS statement 130
and the XBMID option 129
software requirements 80