RECOVER PLUS for DB2
Reference Manual

Supporting

Version 11.2.00 of RECOVER PLUS for DB2
Version 11.2.00 of Administrative Assistant for DB2
Version 11.2.00 of Database Administration for DB2
Version 11.2.00 of Recovery Management for DB2
Version 11.2.00 of BMC Recovery for DB2

May 2015
**Contacting BMC Software**

Several methods are available for contacting BMC Software.

You can access the BMC Software website at [http://www.bmc.com](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>
<tbody>
<tr>
<td>BMC SOFTWARE INC&lt;br&gt;2101 CITYWEST BLVD&lt;br&gt;HOUSTON TX 77042-2827 USA</td>
<td>1 713 918 8800&lt;br&gt;or&lt;br&gt;1 800 841 2031</td>
<td>1 713 918 8000</td>
</tr>
</tbody>
</table>

**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, DB2 Universal Database, FlashCopy, IBM, IBM z, MVS, QMF, RACF, System z, and z/OS 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 UNPUBLISHED—RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES.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**

**Support website**
You can obtain technical support from BMC 24 hours a day, 7 days a week at [http://www.bmc.com/support](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 problems similar to yours and possible solutions
- Order or download product documentation
- Download products and maintenance
- Report a problem or ask a question
- Subscribe to receive proactive e-mail alerts
- Find worldwide BMC support center locations and contact information, including e-mail addresses, fax numbers, and telephone numbers

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

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

- **Product information**
  - Product name
  - Product version (release number)
  - License number and password (trial or permanent)
- **Operating system and environment information**
  - Machine type
  - Operating system type, version, and service pack or other maintenance level such as PUT or PTF
  - System hardware configuration
  - Serial numbers
  - Related software (database, application, and communication) including type, version, and service pack or maintenance level
- **Sequence of events leading to the problem**
- **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
  - Messages from related software
License key and password information

If you have questions about your license key or password, contact Customer Support through one of the following methods:

- Send an e-mail message to customer_support@bmc.com. (In the Subject line, enter SupID:yourSupportContractID, such as SupID:12345.)
- In the United States and Canada, call 1 800 537 1813. Outside the United States and Canada, contact your local support center for assistance.
## Contents

### About this book
- Related publications .............................................. 15
- Conventions ...................................................................... 16
- Syntax diagrams .............................................................. 16
- Summary of changes ...................................................... 18
  - Version 11.2.00 May 2015 ........................................ 18
  - Version 11.1.00 June 2013 ......................................... 20
  - Version 10.1.00 April 2011 ...................................... 25

### Chapter 1 Introduction to RECOVER PLUS
- Overview ........................................................................ 33
- Solution integration ....................................................... 34
- Features and benefits of RECOVER PLUS ....................... 35
  - Reduced elapsed time required to recover .................. 35
  - Better audit and recovery process control .................. 36
  - Expanded recovery capabilities .................................. 37
  - Recovery test capability ............................................ 39
  - TRANSFORM and High-speed Structure Change ............ 39
- RECOVER PLUS operation ............................................ 40
  - Using IBM DB2 resources .......................................... 42
  - Using RECOVER PLUS with RECOVERY MANAGER .... 43
  - Using RECOVER PLUS with R+/CHANGE ACCUM ....... 43
- RECOVER PLUS limitations ........................................... 44
- Planning recovery ......................................................... 46
- Recovery strategies ....................................................... 47
  - Using LOGSCAN for estimates ................................... 47
  - Using BACKOUT for point-in-time recovery without image copies ................................. 48
  - Comparing BACKOUT performance ........................... 49
  - Using BACKOUT AUTO recovery .............................. 50
  - Using copies and logs for index recovery .................... 51
  - Performing forward recovery without an image copy .... 51
  - Recovering with Instant Snapshot copies ................... 52
  - Using parallel MERGE phases and log sorts ................ 52
  - Using multitasking index key sorts and index rebuilds .... 52
  - Eliminating interim data sets for index keys ............... 53
  - Rebuilding nonpartitioned indexes on a partitioned table space ................... 53
### Chapter 3 RECOVER PLUS syntax

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>87</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>87</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>87</td>
</tr>
<tr>
<td>RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE</td>
<td>88</td>
</tr>
<tr>
<td>REBUILD INDEX</td>
<td>88</td>
</tr>
<tr>
<td>RECOVER UNLOADKEYS</td>
<td>89</td>
</tr>
<tr>
<td>RECOVER BUILDINDEX</td>
<td>89</td>
</tr>
<tr>
<td>LOGSCAN</td>
<td>90</td>
</tr>
<tr>
<td>MIGRATE</td>
<td>90</td>
</tr>
<tr>
<td>IMPORT</td>
<td>90</td>
</tr>
<tr>
<td>Adding comments</td>
<td>90</td>
</tr>
<tr>
<td>Use of long names</td>
<td>91</td>
</tr>
<tr>
<td>Support for Unicode names</td>
<td>91</td>
</tr>
<tr>
<td>Using multiple commands</td>
<td>92</td>
</tr>
<tr>
<td>General restrictions for multiple commands</td>
<td>92</td>
</tr>
<tr>
<td>Restrictions for STOGROUP-defined objects</td>
<td>94</td>
</tr>
<tr>
<td>General information about SORTDEVT and SORTNUM</td>
<td>94</td>
</tr>
<tr>
<td>Restrictions for key work data sets, SKEYDDN, SORTNUM, and SORTDEVT</td>
<td>95</td>
</tr>
<tr>
<td>Alphabetical listing of RECOVER PLUS options</td>
<td>96</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>113</td>
</tr>
<tr>
<td>OPTIONS syntax</td>
<td>114</td>
</tr>
<tr>
<td>OPTIONS option descriptions</td>
<td>118</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>151</td>
</tr>
<tr>
<td>OUTPUT syntax</td>
<td>152</td>
</tr>
<tr>
<td>OUTPUT command dependencies and prohibitions</td>
<td>153</td>
</tr>
<tr>
<td>OUTPUT option descriptions</td>
<td>153</td>
</tr>
<tr>
<td>RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE</td>
<td>161</td>
</tr>
<tr>
<td>RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax</td>
<td>161</td>
</tr>
<tr>
<td>RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX command dependencies and prohibitions</td>
<td>174</td>
</tr>
<tr>
<td>RECOVER INDEX and RECOVER INDEXSPACE command dependencies and prohibitions</td>
<td>175</td>
</tr>
<tr>
<td>RECOVER TABLESPACE RECOVER INDEX and RECOVER INDEXSPACE option descriptions</td>
<td>177</td>
</tr>
<tr>
<td>Copy data set ddname construction for JCL-allocated data sets</td>
<td>222</td>
</tr>
</tbody>
</table>
Using MSGLEVEL to control RECOVER PLUS output ..........................................348
   Sample output with MSGLEVEL(0) ...............................................................349
   Sample output with MSGLEVEL(1) ...............................................................353
   Sample output with MSGLEVEL(2) ...............................................................356
Examples of RECOVER PLUS operations .................................................................366
   Multiple table space recovery and index rebuilds ..........................................367
   Parallel index key sorts and multitasking index rebuilds .............................369
   Multiple log sorts, MERGE phases, and parallel index key sorts with
   multitasking index rebuilds ...........................................................................371
   Multiple table space and index recoveries ..................................................373
   BACKOUT recovery .....................................................................................374
   Instant Snapshot recovery ............................................................................375

Chapter 5     Examples of RECOVER PLUS jobs 377
Example 1: Previewing the recovery plan .................................................................384
Example 2: Recovering to the current state ............................................................389
   Example 2A: Using REBUILD INDEX ..........................................................389
   Example 2B: Using RECOVER INDEX .........................................................394
Example 3: Creating and dynamically allocating output copies ..............................410
Example 4: Recovering to a specified copy ............................................................410
   Example 4A: Using TOCOPY with a named copy ........................................411
   Example 4B: Using TOCOPY LASTCOPY ....................................................411
Example 5: Recovering to a specific log point .......................................................413
   Example 5A Using the TOLOGPOINT option with RECOVER ....................414
   Example 5B: Using TOLOGPOINT and BACKOUT ....................................414
Example 6: Extracting nonpartitioned index keys .................................................415
Example 7: Building a nonpartitioned index ..........................................................415
Example 8: Recovering using tape-stacked data sets .............................................416
   Example 8A: Making stacked image copies ...............................................416
   Example 8B: Recovering with stacked image copies ....................................417
Example 9: Recovering to a non-DB2 data set ......................................................417
   Directing recovery output ............................................................................418
Example 10: Overriding installation options ..........................................................428
   Example 10A: Overriding several installation options .................................429
   Example 10B: Overriding the IXRECP installation option ............................429
Example 11: Recovering using a change accumulation file ....................................436
   Example 11A: Creating the change accumulation file ................................436
   Example 11B: Using the change accumulation file .......................................436
Example 12: Using the MAINT parameter ...........................................................437
Example 13: Using the LOGSCAN command ......................................................437
Example 14: Simulating recovery ........................................................................440
Example 15: Using MAXKSORT for parallel index rebuilds ........................................... 441
Example 16: Using MAXKSORT and recovering a table space ......................................... 450
Example 17: Using timestamp recovery ........................................................................... 450
Example 18: Using the MAXLSORT default value ............................................................ 451

Chapter 6  RECOVER PLUS concepts  467

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recovery of partitioned table spaces and indexes</td>
<td>467</td>
</tr>
<tr>
<td>Recovering multiple table space partitions in separate jobs</td>
<td>467</td>
</tr>
<tr>
<td>Recovering table space partitions and rebuilding a partitioned index in multiple jobs</td>
<td>468</td>
</tr>
<tr>
<td>Rebuilding nonpartitioned indexes with single and multiple jobs</td>
<td>469</td>
</tr>
<tr>
<td>Recovering a partitioned table space and rebuilding a nonpartitioned index</td>
<td>471</td>
</tr>
<tr>
<td>Recovering a table space and rebuilding a partitioned index and a nonpartitioned index</td>
<td>472</td>
</tr>
<tr>
<td>Understanding the LOGSORT strategy</td>
<td>473</td>
</tr>
<tr>
<td>Why is it faster to sort the log?</td>
<td>473</td>
</tr>
<tr>
<td>What are the restrictions when using LOGSORT?</td>
<td>474</td>
</tr>
<tr>
<td>Does LOGSORT have other advantages or disadvantages?</td>
<td>474</td>
</tr>
<tr>
<td>Using LOGONLY or LOGAPPLY ONLY with point-in-time recoveries</td>
<td>475</td>
</tr>
<tr>
<td>Understanding the NOWORKDDN strategy</td>
<td>477</td>
</tr>
<tr>
<td>Understanding the UNLOADKEYS and BUILDINDEX strategy</td>
<td>478</td>
</tr>
<tr>
<td>Using SHRLEVEL CHANGE full copies for recovery</td>
<td>479</td>
</tr>
<tr>
<td>Attempting to recover to a SHRLEVEL CHANGE copy</td>
<td>479</td>
</tr>
<tr>
<td>Fallback recovery with RECOVER PLUS</td>
<td>479</td>
</tr>
<tr>
<td>Strategies for point-in-time recovery</td>
<td>481</td>
</tr>
<tr>
<td>Using BACKOUT</td>
<td>481</td>
</tr>
<tr>
<td>Using Instant Snapshots</td>
<td>484</td>
</tr>
<tr>
<td>Changing SHAREOPTIONS for striped or extended format data</td>
<td>488</td>
</tr>
<tr>
<td>Working with ASCII and Unicode data</td>
<td>488</td>
</tr>
<tr>
<td>Managing sort performance</td>
<td>489</td>
</tr>
<tr>
<td>Recommendations for a large number of concurrent sorts</td>
<td>490</td>
</tr>
<tr>
<td>Sort file size estimation</td>
<td>491</td>
</tr>
<tr>
<td>RECOVER PLUS sort parameters</td>
<td>493</td>
</tr>
<tr>
<td>Allocation of sort work space</td>
<td>499</td>
</tr>
<tr>
<td>Strategies for using copies</td>
<td>500</td>
</tr>
<tr>
<td>Image copy data set contents</td>
<td>500</td>
</tr>
<tr>
<td>Cabinet copy data set contents</td>
<td>500</td>
</tr>
<tr>
<td>Using copies containing multiple data sets</td>
<td>500</td>
</tr>
<tr>
<td>Using data sets stacked on tape</td>
<td>501</td>
</tr>
<tr>
<td>Using image copies and log records for indexes</td>
<td>502</td>
</tr>
</tbody>
</table>
Chapter 7  Recovering a dropped table space or table 539

Overview .......................................................................................................................... 539
Example 1: Recovering a dropped table space when image copies are available ........... 540
Example 2: Recovering a dropped table in a segmented table space containing multiple tables ................................................................. 545
Example 3: Recovering a dropped table from a simple table space .............................. 549
Example 4: Recovering a dropped table space using a pack backup .............................................. 551

---

Chapter 8 Migrating data ........................................................................................................... 555
Overview .................................................................................................................................. 555
Defining target objects for migration ....................................................................................... 557
Example 1: Creating a consistent database for query and review ........................................... 558
Example 2: Moving an application to another subsystem with index recovery ....................... 564
Example 3: Moving an application to another subsystem with index rebuilding ......................... 572

---

Chapter 9 Making copies from the log and earlier copies ................................................................ 581
Overview .................................................................................................................................. 581
Example 1: Creating a consistent set of copies at a point in time ........................................... 582
Example 2: Creating an image copy without accessing the space ............................................ 584

---

Appendix A RECOVER PLUS installation options ........................................................................ 587
Installation options macro listing ............................................................................................. 588
RECOVER PLUS installation options ......................................................................................... 589
  Basic installation options ........................................................................................................ 592

---

Appendix B BMCSORT installation and options ........................................................................... 617
Installation overview .................................................................................................................. 617
DYNALOC installation option ..................................................................................................... 617

---

Appendix C Common utility tables ............................................................................................. 621
Overview of common utility tables ............................................................................................ 621
  Warnings and considerations for common utility tables ........................................................ 623
  Managing common utility tables ............................................................................................. 625
BMCDICT table ......................................................................................................................... 626
  BMCDICT table considerations ............................................................................................... 627
  Maintaining the BMCDICT table ............................................................................................ 627
BMCHIST table .......................................................................................................................... 628
  BMCHIST table considerations for COPY PLUS ................................................................... 630
  BMCHIST table considerations for RECOVER PLUS ........................................................... 630
  Maintaining the BMCHIST table ............................................................................................ 631
BMCLGRNX table ...................................................................................................................... 631
BMCSYNC table .......................................................................................................................... 632
  BMCSYNC table considerations ............................................................................................ 635
  Maintaining the BMCSYNC table .......................................................................................... 637
Cleaning up RECOVER UNLOADKEYS entries ...................................................................... 637
Shared access levels of BMC utilities ....................................................................................... 638
BMCTRANS table  .........................................................................................................640
BMCUTIL table  ............................................................................................................. 642
  Maintaining the BMCUTIL table .............................................................................. 645
BMCXCOPY table  .........................................................................................................645
  Maintaining the BMCXCOPY table .........................................................................652

Appendix D         BMC Common DB2 repository tables 653
Naming conventions .................................................................................................... 653
OBJSETS table .................................................................................................................653
OBJSET_DEF table .........................................................................................................655
OBJSET_SQL table .........................................................................................................657
GRPOPTS table ............................................................................................................. 658
PRODREG table ............................................................................................................ 658
GROUPAUTH table ...................................................................................................... 658

Appendix E         RECOVER PLUS syntax diagrams 661
Alphabetical listing of RECOVER PLUS options .................................................................661
OPTIONS command syntax diagram ................................................................................678
OUTPUT command syntax diagram .............................................................................. 683
RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE command syntax diagram .................................................................................................................................684
REBUILD INDEX command syntax diagram .................................................................. 697
RECOVER UNLOADKEYS command syntax diagram ...................................................... 699
RECOVER BUILDINDEX command syntax diagram ....................................................... 701
LOGSCAN command syntax diagram .......................................................................... 701
MIGRATE command syntax .......................................................................................... 705
IMPORT command syntax ............................................................................................ 708
About this book

This book contains detailed product information and is intended for system administrators and 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 support website at http://www.bmc.com/support.

Note

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

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

Related publications

From the BMC Support Central website, you can use the following methods to access related publications that support your product or solution:

View Quick Course videos (short overviews of selected product concepts, tasks, or features), which are available from the following locations:

— Documentation Center

— Support Central (at http://www.bmc.com/support/mainframe-demonstrations)

— BMC Mainframe YouTube channel (https://www.youtube.com/user/BMCSoftwareMainframe)


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

Tip
You can access the BMC Support Central site at http://www.bmc.com/support.

Conventions

This document 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

■ Menu sequences use a symbol to convey the sequence. For example, Actions => Create Test instructs you to choose the Create Test command from the Actions menu.

Syntax diagrams

The following figure shows the standard format for syntax diagrams:
The following example illustrates the syntax for a hypothetical DELETE statement. 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 option.
- If a diagram shows punctuation marks, parentheses, or similar symbols, you must enter them as part of the syntax.
- In general, IBM commands, keywords, clauses, and data types are displayed in uppercase letters. However, if an item can be shortened, the minimum required
portion might be shown in uppercase letters, with the remainder in lowercase (for example, CANcel).

- The following conventions apply to variables in syntax diagrams:
  - Variables are typically 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).

### 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 11.2.00 May 2015

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

- **End of support for DB2 Version 9**
  Starting with this release, RECOVER PLUS does not support IBM DB2 Version 9. Earlier releases will continue to support Version 9. In addition, RECOVER PLUS supports DB2 Version 10 only in new-function mode.

- **Support tagged copies for migration**
  In the previous release, when EXPORT was run to identify copies for migration, it could be confusing as to which copies were exported, especially if the copies were made in multiple jobs. This release allows RECOVER PLUS to "tag" output copies to simplify selecting copies for migration.
  See TAG tagName in “OUTCOPY specification” on page 212.

- **Migrate XML spaces**
  This release adds support for migrating XML spaces using image copies.

- **Migrate statistics**
  This release adds support for migrating statistics. See “STATS” on page 315 in the IMPORT COMMAND.
- Allow OUTCOPY ONLY using inline copies
  This release adds support for making output copies using inline input copies.

- Allow OUTCOPY ONLY using Instant Snapshots
  This release adds support for making output copies using Instant Snapshot input copies.

- REBUILD INDEX SHRLEVEL CHANGE
  Starting with this release, RECOVER PLUS supports rebuilding indexes with SHRLEVEL CHANGE. See “SHRLEVEL” on page 242 in the REBUILD INDEX command.

- Allow incremental copies with Instant Snapshots
  This release adds support for using incremental copies based on full Instant Snapshot copies for recovery.

- Online Consistent Copies for LOBs
  Starting with this release, RECOVER PLUS can recover LOB spaces using Online Consistent Copies.

- Support for CHANGE ACCUM on DB2 Version 11
  This release of RECOVER PLUS supports CHANGE ACCUM on DB2 Version 11.

- Recover individual spaces using system backups
  This release adds support for recovering individual spaces using system backups. See “RESOURCE SELECTION” on page 126.

- Instantiate data set for DEFINE NO
  When migrating data to a space that is DEFINE NO, starting with this release RECOVER PLUS will update the DB2 catalog and create the cluster if it does not exist. “DEFINE” on page 315.

- Migration of Partition by Growth (PBG) spaces with more than 1 data set
  Using the MIGRATE command, if the target space has more than 1 data set, RECOVER PLUS will automatically add the data sets to the target system.

- Support for not indexing keys that are NULL
  Starting with this release, RECOVER PLUS can omit null keys during REBUILD INDEX.

- Support RBA/LRSN conversion during REBUILD INDEX
  Starting with this release, RECOVER PLUS can convert the RBA/LRSN format (basic-extended) during REBUILD INDEX. See “RBALRSN_CONVERSION” on page 245.
- Support RBA/LRSN conversion for point-in-time recovery
  Starting with this release, RECOVER PLUS allows recovery to a point in time
  where the RBA/LRSN format for the partition is different than the current RBA/
  LRSN format.

- Recover to a point in time before DROP COLUMN
  This release of RECOVER PLUS checks for DROP COLUMNs and disallows
  recovery to a point in time before a DROP COLUMN.

- Recover to a point in time before materializing changes
  Starting with this release, recovery to a point in time before certain materializing
  changes (e.g. DSSIZE, PGXIZE, and SEGSIZE) is allowed. See “Materialized
  pending changes” on page 520.

- Allow INDEP OUTSPACE with OBJECTSET
  Starting with this release, RECOVER PLUS allows the use of INDEP OUTSPACE
  with the RECOVER and REBUILD commands when an object list is specified via
  OBJECTSET syntax.

- New RICHK option
  If RICHK YES is specified and a partial recovery is performed on space(s) with
  referential integrity constraints, RECOVER PLUS will set check pending on
  related spaces that were not recovered. See RICHK in the OPTIONS (“RICHK” on
  page 128), IMPORT (“TABLESPACE specification” on page 304), and RECOVER
  (“Table space specification” on page 178) commands, and as an installation
  option “RICHK=NO” on page 615.

- New AUX ALL option
  In previous releases, AUX YES would recover history tables as well as the
  auxiliary spaces. This release separates history tables using the HISTORY option
  and adds AUX ALL to recover all of the auxiliary spaces. See “AUX” on page 128.

- Support for Persistent Read Only (PRO) and read-or-replication-only (RREPL)
  statuses
  This release adds support for PRO and RREPL statuses.

### Version 11.1.00 June 2013

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

End of support for DB2 Version 8
Starting with this release, RECOVER PLUS does not support IBM DB2 Version 8. Earlier releases will continue to support Version 8.

Extended RBA and LRSN support

All RECOVER PLUS output has been converted to show 10-byte RBAs and LRSNs. The BMC utilities database has also changed to support extended RBAs and LRSNs.

New IMPORT and MIGRATE commands for migrating data

New IMPORT and MIGRATE commands allow you to move data from one or more table spaces to another via a migration file that the COPY PLUS EXPORT command creates. The IMPORT command uses copies as input; the MIGRATE command uses copies and log. For more information, see the MIGRATE command description and the IMPORT command description.

Note

This feature requires one of the following valid passwords:

- Recovery Management solution password
- Database Administration solution password

The Database Administration (DAD) password allows RECOVER PLUS to run within a worklist, but does not include full-function RECOVER PLUS.

This release also includes the following changes:

- Adds new EXPSSID, EXPSLRSN, and EXPTLRSN columns to the BMCXCOPY table to accommodate the EXPORT, MIGRATE, and IMPORT commands
- Adds ICTYPE column value m, and COPY_TYPE column values X and I to the BMCXCOPY table

In addition to the references above, see the following resources for more information:

- "Migrating or importing data"
- EXPORT command description in the COPY PLUS for DB2 Reference Manual
- "Moving data with a migration file," in Recovery Management for DB2 User Guide

Transformation restrictions removed

Transformation restrictions have been removed from High-speed Structure Change, as follows:
Multi-table table space transformations are now allowed. Previously, the source table space could contain only one table.

BRF to RRF and RRF to BRF transformations are now allowed. Previously, the source and target table space had to have the same row format. You could not transform a table space with basic row format (BRF) into a table space that had reordered row format (RRF), or vice versa.

**Note**
The rows are not transformed, but the DB2 catalog is changed to reflect the row format from the source.

Transformations to and from MEMBER CLUSTER are now allowed. Previously, Recovery Management was unable to handle transformations related to the MEMBER CLUSTER attribute.


### Additional DSSIZE values

The following values are now valid for DSSIZE:

- 128G
- 256G

### Estimation improvements

Estimation has been improved to better account for current hardware and multitasking. Additionally, the following new installation options accommodate estimation:

- DISKIORATE specifies the number of megabytes per second that RECOVER PLUS reads from disk. The default value is 100.

- CPUMIPS specifies the rate at which a CPU executes instructions, in millions of instructions per second (MIPS). The default value is 200.

Many of the performance factors used in estimation are derived from DISKIORATE and CPUMIPS.

Estimation requires a Recovery Management *for DB2* solution password. For more information, see "Recovery simulation and estimation," in the *Recovery Management for DB2 User Guide*.

### Use of symbolics to INCOPY MODEL *dataSet*Name syntax
The data set specified with INCOPY MODEL syntax supports the use of symbolics.

**New symbolic variables**

RECOVER PLUS adds the &UNIQ (or &UQ) symbolic variable to generate unique data set names. For example, you can use this variable with the DSNAME parameter on the OUTPUT command.

If you use &UNIQ, RECOVER PLUS generates a 1- to 8-character value that is based on the system clock. The first character is always an uppercase letter. Each remaining character is either an uppercase letter or a numeral from 0 through 9.

RECOVER PLUS also adds the &PART5 symbolic variable that you can use for any data set. RECOVER PLUS generates 5-character partition numbers as follows:

- Partition 1 = 00001
- Partition 10 = 00010
- Partition 100 = 00100
- Partition 1000 = 01000
- Nonpartitioned = 00000

**Allow INCOPY with OBJECTSET**

The restrictions that did not allow INCOPY with OBJECTSET have been removed.

**Automatic sizing of dynamically allocated output copies on DASD**

RECOVER PLUS now automatically sizes dynamically allocated output copies on DASD when the primary allocation quantity is not specified by the SPACE option.

To accomplish automatic sizing, RECOVER PLUS adds the MAXPRIM option to the OUTPUT command. MAXPRIM lets you set a maximum amount of disk space (in the units specified by SPACE) to allocate as primary space. Valid values are 0 through 65535. A nonzero value for MAXPRIM establishes an upper limit for primary space allocation; 0 specifies no limit.

**New EXCLUDE clause for OBJECTSET**

For the RECOVER and REBUILD INDEX commands, RECOVER PLUS adds the EXCLUDE option to the OBJECTSET specification.

**Support for VSAM data sets on nonsnappable disks**

When you use the new installation option SNAP, RECOVER PLUS can now recover VSAM copies, even if the data set is not on a snappable disk. SNAP is also available on the OPTIONS command. When the SNAP value is VSAM and the data set is not on a snappable disk, RECOVER PLUS reads a VSAM data set using the INCOPY
FULL SNAPSHOT DSNAME dataSetName syntax, and can also recover using a 
VSAM data set registered in BMCXCOPY or SYSCOPY.

Additionally, RECOVER PLUS adds the FCPPRC installation option to control what 
happens if you specify a value of VSAM with SNAP and the data sets are on a disk 
that is capable of IBM FlashCopy.

**New AFRPRnnn data sets to eliminate deferred messaging**

This release removes deferred messaging for subtasks. Now, each execution phase 
has its own AFRPRnnn print file, and messages are displayed in the print file as soon 
as they are issued. The maximum number of print files allowed by RECOVER PLUS 
is calculated by the following formula:

\[
\text{value of MAXKSORT} + \text{value of MAXLSORT} + 1 = \text{maximum number of AFRPR nnn files}
\]

The print files are named AFRPR001, AFRPR002, AFRPR003, and so on. AFRPRINT 
now displays the name of the AFRPR nnn file used by each phase for each object.

**New USELOGS installation option**

The new USELOGS installation option lets you specify the order in which to read 
active and archive log files. USELOGS=(ACT1,ACT2,ARC1,ARC2) is the default 
value.

You can override this installation option by using the RESOURCE SELECTION 
LOGS syntax on the OPTION command.

**Better handling of partition-by-growth migrations**

This release provides better handling for partition-by-growth (PBG) migrations. This 
release removes the warning against PBG migrations that have a different number of 
source and target data sets.

**Recovery of an individual space from a system-level volume backup**

To recover an individual space from a system-level volume backup, you can use 
COPY IMAGECOPY in the BMC COPY PLUS product to make a standard copy of 
the space. Then use RECOVER PLUS to recover the space from the standard copy.

**Documentation changes**

This release includes the following documentation changes:

- The Point-in-time recovery specification has been renamed to the TORBA/ 
  TOLOGPOINT specification in "RECOVER PLUS syntax" and in "RECOVER 
  PLUS syntax diagrams."
All messages are now available in the BMC Documentation Center, which is accessible from the BMC Support Central site (http://www.bmc.com/support). A separate messages manual is no longer available.

Installation and configuration information is now located in the following books:

— *Installation System User Guide*

— *BMC Products and Solutions for DB2 Configuration Guide*

**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.

■ 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 supports FlashCopy. FlashCopy is also supported in the installation options LOCCPSEL and REMCPSEL 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. Also, updates to the DSNNAME description of the INCOPY specification 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>ENFM8 a</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>ENFM9 a</td>
<td>enabling-new-function mode from DB2 Version 9</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.

<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>

1. greater timestamp precision (extends microseconds to 12 places, but 6 remains the default)

2. **TIMESTAMP WITH TIME ZONE** data type

3. 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, and EATTR, STORCLAS and DATAACLAS 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.

4. last/currently committed data

   RECOVER PLUS supports the new log records that this feature generates.

5. temporal tables

   To support temporal tables, RECOVER PLUS adds the HISTORY keyword to the AUX installation option. 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

   — RECOVER TABLESPACE

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

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 option enables this functionality. For more information, see "ZIIP."

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."

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. 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 and as an installation option. 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 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` and PART `begin:end` syntax.

**NUMREC changes for key sorts**

For key sorts only, RECOVER PLUS has changed some NUMREC parameters for REBUILD INDEX and RECOVER UNLOADKEYS:

- 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. 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

**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.
Introduction to RECOVER PLUS

This section describes the features and advantages of using RECOVER PLUS.

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
- 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 “Recovering a dropped table space or table” on page 539.

With the Copy Migration feature of the Recovery Management solution, RECOVER PLUS will obtain the object ID values from a migration file and either the source or target catalog, simplifying the migration process. For more information, see the Recovery Management for DB2 User Guide.

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 BMCXCOPY
- 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

With the Copy Migration feature of the Recovery Management solution, RECOVER PLUS will obtain the input image copy name from a migration file or the source catalog, simplifying the migration process. For more information, see the Recovery Management for DB2 User Guide.

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 partition-by-growth (PBG) table space
- non-large partitioned table space to large partitioned table space
- range-partitioned table space to range-partitioned (PBR) universal table space
- range-partitioned table space to partition-by-growth universal table space
- multi-table table space to partition-by-growth universal table space

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:

- change the segment size (SEGSIZE) of a table space
- transform a non-segmented table space into a segmented one
- change the data set size (DSSIZE or PIECESIZE) of your table space or index
- change the compression attribute of your index
- add or remove the MEMBER CLUSTER attribute
- increase or decrease MAXPARTITIONS

RECOVER PLUS performs the page transformations and uses the TRANSFORM option ("Table space specification" on page 178), the DSSIZE option ("INCOPY specification" on page 217), and the PIECESIZE option ("INCOPY specification" on page 217).

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 and COPY 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)
- migration file

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 “RECOVER PLUS syntax” on page 87.

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 9 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

■ 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.

■ 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 DSNZPARN 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:

■ RECOVER PLUS calls DSNUTILB for the following DB2 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, LOWER, DAY, MONTH, and YEAR.

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.

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

■ RECOVER PLUS versions earlier than 11.1.00 will not work with DB2 Version 11.

■ If you recover an object to a point in time using 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

**Note**

Do not use RECOVER PLUS to back out over a range that includes an IBM backout recovery. Do not use IBM recover to recover over a range that includes a RECOVER PLUS backout.

**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.
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 34. 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 (“INDEXLOG=NO” on page 594).

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 261 and “Example 13: Using the LOGSCAN command” on page 437.

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 51.)

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 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
  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.) COPY NO indexes are eligible for BACKOUT processing.

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.

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 481.

### Comparing BACKOUT performance

The backout recovery strategy is dramatically faster than most traditional forward recoveries as shown in the figure below. 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 (“BACKOUT=AUTO” on page 609) 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 (“BACKOUT” on page 131) on the OPTIONS command in the JCL.

---

**Note**

BACKOUT to forward recovery is a feature of the Recovery Management solution and the BACKOUT AUTO option requires the use of a valid Recovery Management solution password. For more information about BACKOUT AUTO recovery and Recovery Management, see the *Recovery Management for DB2 User Guide*.

---

**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 SYSIBM.SYSCOPY as IBM FlashCopy (FC) copies if they are LP copies and in the BMCXCOPY table otherwise. 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 484.

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 (“MAXLSORT integer ” on page 145 and “MAXLSORT=0” on page 606). 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” on page 52) with log sorts, the KSORTSHARE option (“KSORTSHARE ” on page 146 and “KSORTSHARE=YES” on page 607) 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 497.

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 (“MAXKSORT integer ” on page 143 and “MAXKSORT” on page 597). 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” on page 52 can also improve your index recovery.

For more information, see “Setting the MAXKSORT option” on page 495.

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
- SYSTEM BACKUP
- FlashCopy (image copies only)

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 to go a step beyond previewing recovery activity to testing it.

This non-destructive option (“SIMULATE YES” on page 141) 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 (“SIMRCVR” on page 177)
- SIMRBLD INDEX for REBUILD INDEX (“SIMRBLD” on page 227)
- SIMRCVR UNLOADKEYS for RECOVER UNLOADKEYS (“RECOVER UNLOADKEYS” on page 245)
- SIMRCVR BUILDINDEX for RECOVER BUILDINDEX (“RECOVER BUILDINDEX” on page 254)

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 521.

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 63.

Migration files which are created by the COPY PLUS EXPORT command are registered in BMCXCOPY.

The BMCXCOPY table is described in “BMCXCOPY table” on page 645.

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.

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 ROWID1 = ROWID(X'50633BFCD66BD08B210401581C30404040')
---------+---------+---------+---------+---------+---------+-----
EMPNO  FNAME                 LNAME
---------+---------+---------+---------+---------+---------+-----
1  JOHN                  SMITH
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

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 CKP 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 479.

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
- 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 82.

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:

```
RECOVER INDEX IX.A
```

or

```
RECOVER INDEX IX.A DSNUM  ALL
```

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

```
RECOVER INDEX IX.A DSNUM  n
```

or

```
RECOVER INDEX IX.A DSNUM  n:n(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. 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).

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 ("DATAMVR=" on page 606) or specify the DATAMVR option on the OPTIONS statement ("DATAMVR programName " on page 138) 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.
Using a system-level volume backup

DB2 supports the BACKUP SYSTEM command, which backs up the entire DB2 system using volume snapshots. To recover an individual space from a system-level volume backup, you can use RECOVER in RECOVER PLUS and specify SB (for system backup) in the RESOURCE SELECTION for image copies. Also, COPY IMAGECOPY in COPY PLUS can make a standard copy of the space from a system backup.

For more information about using the COPY IMAGECOPY to create a standard copy from a system-level volume backup, see Chapter 2 of the COPY PLUS Reference Manual.

Migrating or importing data

With the Copy Migration feature of the Recovery Management solution, you can use the RECOVER PLUS MIGRATE and IMPORT commands, along with the migration file from the COPY PLUS EXPORT command, for simplified data migrating or importing. The process includes the following benefits:

- supports importing copies that use BMC proprietary formats
- does not require that you manage data sets to keep up with current copies
- does not require that you know the OBIDs from the source system
- imports applications and object sets as a single unit
- does not require the import of unchanged objects (to save time)
- supports the use of older EXPORT files to back-date imports
- provides a less error prone, and therefore, more accurate data migration (for example, less likely to use the wrong data set name or OBIDs)

Note

To receive all of the benefits provided above, you must have one of the following valid passwords:

- a Recovery Management solution password
- a Database Administration solution password

For more information, see the following sources:

- the Recovery Management for DB2 User Guide
The EXPORT command description in the *COPY PLUS for DB2 Reference Manual*

“MIGRATE” on page 277 and “IMPORT” on page 297

**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 528.)
Operational considerations

This section discusses issues and concepts you should consider when using the RECOVER PLUS product in an operational environment.

Operating environment

RECOVER PLUS requires the operating environment described in this section.

DB2 support

This version of RECOVER PLUS supports DB2 Versions 10 and 11.

System requirements

This version of RECOVER PLUS supports IBM-supported versions of z/OS that are compatible with DB2 Version 10 or 11.

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 2 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 SMFPRM xx 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 11.1.00 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 6.2 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.

  **Note**
  
  If RECOVER PLUS invokes DSNUTILB, the call displays in the AFRPLAN and AFRPRINT output files, and DSNUTILB messages display in the AFRPRnnn output files.

**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.

**Authorization verification mechanisms**

If the DB2 DSNX@XAC authorization exit is available for your system, RECOVER PLUS uses this exit to verify authorization for external access. The exit is available from the following sources:
IBM provides a sample exit with DB2 for the IBM Resource Access Control Facility (RACF) component.

CA Technologies provides the DSNX@XAC exit with CA-ACF2 Security for DB2 and CA-Top Secret Security for DB2.

BMC recommends this mechanism for implementing external security. The access control authorization exit must be available in the STEPLIB, JOBLIB, linklist, or in the SYS3.DSN exit.

If the DSNX@XAC exit is not available, RECOVER PLUS uses the standard DB2 method to check security.

**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 on page 73 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 593.

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.
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.

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 “RDB2STAT override parameter (rdb2Stat)” on page 323 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

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

**Table 2: RECOVER PLUS actions for access to DB2 table spaces**

<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 UTR0 AREST RESP REFP b</td>
</tr>
<tr>
<td>RECOVER PLUS command</td>
<td>Action taken by RECOVER PLUS for DB2 table spaces for acceptable initial status</td>
<td>Action taken by RECOVER PLUS for DB2 table spaces for unacceptable initial status</td>
<td>Unacceptable initial table space status</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>----------------------------------------</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 on page 77 shows the action taken by RECOVER PLUS for acceptable and unacceptable initial DB2 index space status.

---

Table space and index space status
<table>
<thead>
<tr>
<th>RECOVER PLUS command</th>
<th>Action taken by RECOVER PLUS for DB2 index spaces for acceptable initial status</th>
<th>Action taken by RECOVER PLUS for DB2 index spaces for unacceptable initial status</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 UTRO AREST RESTP REFP</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 UTRO RBDP PSRCP PSRBD WEPR 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 UTRO AREST RESTP</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>

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.

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 323.

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 74.
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 642.

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 632.

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 on page 79.

The "Access level" column in Table 4 on page 79 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

<table>
<thead>
<tr>
<th>Product</th>
<th>Accesslevel</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 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:

### Table: Access Levels and Additional Information

<table>
<thead>
<tr>
<th>Product</th>
<th>Accesslevel</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>
</tbody>
</table>
| RECOVER PLUS     | X, S, or blank | ■ Under the following conditions, RECOVER PLUS registers an object with shared access (S):  
|                  |             |   — 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.  
|                  |             |   — A table space partition is registered with shared access if the keys for that partition are unloaded with a RECOVER UNLOADKEYS operation.  
|                  |             | ■ If you specify the following commands or options, RECOVER PLUS registers the object with no access status (blank):  
|                  |             |   — the ACCUM command  
|                  |             |   — OUTCOPY ONLY  
|                  |             |   — INDEP OUTSPACE  
|                  |             | ■ In all other cases, RECOVER PLUS registers the object with exclusive access (X). |
| RECOVERY MANAGER | S           | none                                                                                   |
| REORG PLUS       | X           | 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. |
| UNLOAD PLUS      | S           | none                                                                                   |
- 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 92.

**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 “RECOVER PLUS installation options” on page 587.

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 “Building and executing RECOVER PLUS jobs” on page 317.

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.

```plaintext
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
```
Specific volume serial numbers are used in the order in which they are returned by
SQL from SYSIBM.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 123.

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

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

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.

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 “RECOVER PLUS syntax diagrams” on page 661.

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 92.

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 113.

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 151.
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 and RECOVER INDEXSPACE option descriptions” on page 177.

**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 594 and “INDEXLOG” on page 134).

REBUILD INDEX

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 223.

**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 or the RECOVER BUILDINDEX.

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 245.

**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 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 254.

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 261.

MIGRATE

The MIGRATE command allows you to move data from one or more table spaces to another. The table spaces can be on the same subsystem, or they can be on different subsystems, as long as the two subsystems share DASD. Image copies, and optionally DB2 log, from the source system are used to create the target image.

For more information, see “MIGRATE” on page 277.

IMPORT

The IMPORT command allows you to move data, by use of image copies, from one or more table spaces to another.

For more information, see “IMPORT” on page 297.

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*/
```
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 on page 92 shows an example of how RECOVER PLUS represents Unicode in SYSPRINT.

Figure 2: Unicode representation in the RECOVER PLUS output file

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:

```
REBUILD INDEX (AFRIND1)
REBUILD INDEX(ALL) TABLESPACE AFRDB.AFRTS
```

However, the following command statements are correct:

```
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 TABLESPACE AFRDB.AFRTS
```

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 203.

- 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 241.

- 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 260.

General information about SORTDEVT and SORTNUM

You can specify SORTDEVT and SORTNUM in the installation options, 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 NOWORKDDN by default.

Alphabetical listing of RECOVER PLUS options

RECOVER PLUS options are listed in the following table, alphabetized by RECOVER PLUS command, and within the command by option name.
Table 5: RECOVER PLUS command options

<table>
<thead>
<tr>
<th>Command name</th>
<th>Reference</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMPORT</td>
<td>“IMPORT” on page 297</td>
<td>ANALYZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CHANGE ... FROM...TO</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXCLUDE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for OBJECTSET specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for USING specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEPENDENT OUTSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOCOPYPEND</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSCRATCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPYDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REBUILD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REUSE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RICHK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SYNC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UPDATE VERSIONS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>USING</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
<td>---------</td>
</tr>
<tr>
<td>LOGSCAN</td>
<td>“LOGSCAN” on page 261</td>
<td>AUX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BACKOUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMLOGPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint’</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMRBA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint’</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for single index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for multiple index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXES (or INDEX)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for single index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for multiple index specification</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>LOGSCAN, continued</td>
<td>“LOGSCAN” on page 261</td>
<td>LOGAPPLY ONLY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGONLY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TOLOGPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ LASTCOMMONQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ X’logPoint</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TORBA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ LASTCOMMONQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ X’logPoint</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MIGRATE</td>
<td>“MIGRATE” on page 277</td>
<td>ANALYZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CHANGE ... FROM...TO</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXCLUDE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for OBJECTSET specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for USING specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMLOGPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X'logPoint</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMRBA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X'logPoint</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP OUTSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGAPPLY ONLY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGONLY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGSORT</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
<td>---------</td>
</tr>
<tr>
<td>MIGRATE, continued</td>
<td>“MIGRATE” on page 277</td>
<td>MODEL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NUMREC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ABS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ AVGRECSZ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ CALC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ EST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ NOEST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TOCOPY LASTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TOLOCATION</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TOLOGPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOMMONQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE (relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LOGMARK logMarkName (logMarkGeneration)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TORBA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOMMONQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE (relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LOGMARK logMarkName (logMarkGeneration)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td></td>
<td></td>
<td>USING</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>“OPTIONS” on page 113</td>
<td>ANALYZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTOSIZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BACKOUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CHECKINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DATAMVR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EARLYCAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EARLYRECALL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXLOG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IXRECP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>KEYSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>KSORTSHARE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGPOINT X’logPoint’ or LOGPOINT LOGMARK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>logMarkName (logMarkGeneration)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXDRIVES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXKSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXLOGS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXLSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOEARLYCAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOEARLYRECALL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSYSLGRNG</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>OPTIONS, continued</td>
<td>“OPTIONS” on page 113</td>
<td>NUMREC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ ABS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ AVGRECSZ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ CALC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ EST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ NOEST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ON ERROR ANY CONTINUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ ASCODED</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ BYPART</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESINV</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESOURCE SELECTION</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ ACCUMS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ COPIES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ LOGS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RICHK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SIMULATE YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SMCORE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SNAP</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OPTIONS, continued</td>
<td>“OPTIONS” on page 113</td>
<td><strong>SORTDEVT</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- for KEYSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- for LOGSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>SORTDIAG</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>SORTDYN</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>SORTNUM</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- for KEYSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- for LOGSORT</td>
</tr>
</tbody>
</table>
|                       |                                | **STOGROUP**...

**USEORDER**

**TIMESTAMP**

**TRTCH**

**URIDDDN**

**USEACCUM**

**USEHDROBIDS**

**WTOR**

**XBMID**

**ZIIP**
<table>
<thead>
<tr>
<th>Command name</th>
<th>Reference</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>“OUTPUT” on page 151</td>
<td>CATLG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DATACLAS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNAME</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EATTR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXPDT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXPRIM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MGMTCLAS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODELDCB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RETPD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STACK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STORCLAS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRTCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNIT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNITCNT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VOLUMES</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>REBUILD INDEX</td>
<td>“REBUILD INDEX” on page 223</td>
<td>ANALYZE</td>
</tr>
<tr>
<td>SIMRBLD INDEX</td>
<td></td>
<td>AUX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP INTABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP OUTSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for INDEP INTABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for INDEP OUTSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSCRATCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NUMREC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ABS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ CALC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ EST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ NOEST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOWORKDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PART</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REUSE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WORKDDN</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>------------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>RECOVER</td>
<td>“RECOVER BUILDINDEX” on page 254</td>
<td>ANALYZE</td>
</tr>
<tr>
<td>BUILDINDEX</td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td>SIMRCVR</td>
<td></td>
<td>INDEP OUTSPACE</td>
</tr>
<tr>
<td>BUILDINDEX</td>
<td></td>
<td>MODEL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSCRATCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REUSE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEYDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
</tbody>
</table>
### Alphabetical listing of RECOVER PLUS options

<table>
<thead>
<tr>
<th>Command name</th>
<th>Reference</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX</td>
<td></td>
<td>ANALYZE</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>AUX</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>BACKOUT</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>DBID</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>DEFINE</td>
</tr>
<tr>
<td></td>
<td>&quot;RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE&quot; on page 161</td>
<td>DROPRECOVERY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNNAME</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for INDEX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for INDEXSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for TABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSSIZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ENCRYPTED</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMLOGPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMRBA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RECOVER INDEX</td>
<td></td>
<td>INCOPY FULL</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>INCR</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>INDEP OUTSPACE</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>INDEVT</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>INDEX</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>■ for single index specification</td>
</tr>
<tr>
<td>(continued)</td>
<td></td>
<td>■ for multiple index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXES (or INDEX)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for single index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for multiple index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INLINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INLOG LOGPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INLOG RBA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSEQNO</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INVOLUME</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTCOPY (relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOCALSITE</td>
</tr>
</tbody>
</table>

Alphabetical listing of RECOVER PLUS options

Chapter 3  RECOVER PLUS syntax    109
<table>
<thead>
<tr>
<th>Command name</th>
<th>Reference</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX</td>
<td>“RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE” on page 161</td>
<td>LOGAPPLY ONLY</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>LOGONLY</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>LOGPOINT</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>LOGSORT</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>MODEL</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>NOCOPYPEND</td>
</tr>
<tr>
<td>(continued)</td>
<td></td>
<td>NOSCRATCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOWORKDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NUMREC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ABS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ AVGRECSZ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ CALC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ EST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ NOEST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBIDXLAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPYDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPYDSN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PIECESIZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RBA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYDSN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYSITE</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>RECOVER INDEX</td>
<td>&quot;RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE&quot; on page 161</td>
<td>REDEFINE</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>REGISTER</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>RESET</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>RESETRTS</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>REUSE</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>RCHK</td>
</tr>
<tr>
<td></td>
<td>(continued)</td>
<td>SHRLEVEL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SNAPSHOT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for table space specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for multiple index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TOCOPY</td>
</tr>
</tbody>
</table>
### Alphabetical listing of RECOVER PLUS options

<table>
<thead>
<tr>
<th>Command name</th>
<th>Reference</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX</td>
<td>“RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE” on page 161</td>
<td>TOLOGPOINT</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>■ LASTCOMMONQ</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>■ LASTQUIESCE &lt;br&gt; (relativeGenerationNumber)</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>■ LOGMARK logMarkName &lt;br&gt; (logMarkGeneration)</td>
</tr>
<tr>
<td>(continued)</td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td>TORBA</td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOMMONQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE &lt;br&gt; (relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LOGMARK logMarkName &lt;br&gt; (logMarkGeneration)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td>TOSEQNO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOVOLUME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRANSFORM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPDATE VERSIONS</td>
<td></td>
<td></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

The following figures show the syntax of the OPTIONS command.

Figure 3: OPTIONS syntax diagram

1 The default for ANALYZE ONLY runs
Figure 4: OPTIONS syntax diagram, continued

1 Valid with R+/CHANGE ACCUM only
2 Requires a Recovery Management for DB2 solution password
Figure 5: OPTIONS syntax diagram, continued

1 Requires a Recovery Management for DB2 solution password
Figure 6: OPTIONS syntax--LOGSORT specification

LOGSORT specification

Figure 7: OPTIONS syntax--STOGROUP specification

STOGROUP specification

Figure 8: OPTIONS syntax--SORT specification

SORT specification
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 SORDEVT, 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 473.

RECOVER PLUS invokes BMCSORT, which allocates the required temporary sort work data sets. You can perform one of the following actions:

- specify SORTNUM and SORDEVT to direct the allocation
- specify neither SORTNUM nor SORDEVT, 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 SORTDEV and SORTNUM specifications that are present are ignored. If you do not use L001WKnn DD statements and do not specify SORTDEV or SORTNUM, the values that are supplied in the RECOVER PLUS installation options are used.

**SORTDEV deviceType**

SORTDEV 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 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 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 “RECOVER PLUS concepts” on page 467.
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.

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 "TABLESPACE databaseName.tableSpaceName" on page 178). 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 84.

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

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).

```
OPTIONS STOGROUP PRODSG1 USEORDER (PROD01,PROD02,PROD03,PROD04)
STOGROUP PRODSG2 USEORDER (PROD02,PROD03,PROD04,PROD01)
STOGROUP PRODSG3 USEORDER (PROD03,PROD04,PROD01,PROD02)
STOGROUP PRODSG4 USEORDER (PROD04,PROD01,PROD02,PROD03)
```
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 489.)

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

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 (“RESINV=0K” on page 600) 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.

For RECOVER PLUS versions earlier than version 4.1.00, which do not use BMCSORT, the following information may apply:

- If the system sort routine is PLSORT, always set RESINV=0K.
- RESINV corresponds to the SyncSort RSRVTI installation option.
- For DFSORT, RESINV is used only when DFSORT is program-invoked and the DFSORT parameter SIZE=MAX is in effect.

**SMCORE (integerK, integerK)**

This option overrides the SMCORE installation option (“SMCORE=(0K,0K)” on page 599) 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 RECOVER PLUS versions earlier than version 4.1.00, which do not use BMCSORT, the following information may apply:

- For SyncSort, both the first and the second values are used because SyncSort does use memory both above and below the 16 MB line.
- For DFSORT, only the first value is used because DFSORT does not allow separate specification of memory usage below the 16 MB line.
For more information, see “Managing sort performance” on page 489 if your system sort routine is DFSORT, SyncSort, or another package.

**RESOURCE SELECTION**

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

**RESOURCE SELECTION LOGS**

You can specify RESOURCE SELECTION 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.

You use RESOURCE SELECTION LOGS on the OPTIONS command to override the USELOGS installation option (see “USELOGS=(ACT1,ACT2,ARC1,ARC2)” on page 614).

**RESOURCE SELECTION ACCUMS**

You can specify RESOURCE SELECTION 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.

**RESOURCE SELECTION COPIES**

You can specify RESOURCE SELECTION as LP, LB, RP, RB, FC, and SB 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.

SB indicates system backup copies. Similar to SYSTEMPAGES NO copies, if the space is versioned or compressed, any indexes that need to be rebuilt must be rebuilt in a separate step after the table space is recovered. Index recoveries can be done concurrently with table space recoveries. RECOVER PLUS can use only system backups that are on disk.

The default order is FC, LP, LB when operating as a local site, and RP, RB, FC when operating in a recovery site. SB is not included in default processing. You can omit references to copies of the resource that you do not want considered.
**MAXDRIVES integer**

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.

For NOLOGSORT runs, RECOVER PLUS arranges the process to use no more than the MAXDRIVES value. For LOGSORT runs, if more drives are required for the MERGE processing than the MAXDRIVES value, RECOVER PLUS does not start the process. For LOGSORT runs, 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.

**RICHK**

RICHK specifies if RECOVER PLUS will check for referential integrity constraints on the objects being recovered.

**RICHK YES**

RICHK YES checks for referential integrity constraints and sets check pending (CHKP) status for dependent objects not recovered in the same step to the same PIT.

**RICHK NO**

RICHK NO does not check for referential integrity. NO is the default value.

**AUX**

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

If you do not specify AUX on one of these commands, RECOVER PLUS uses the value of the AUX installation option ("AUX=NO" on page 608), 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 ALL, AUX XML, or AUX LOB, you cannot specify the following options:

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

Valid values are NO, ALL, XML, and LOB.

**AUX NO**

RECOVER PLUS does not include XML or LOB objects. You may include HISTORY spaces with the HISTORY keyword when a table space with system-maintained temporal tables is specified in SYSIN. See “HISTORY” on page 130.

**AUX ALL**

RECOVER PLUS includes XML and LOB objects for the base table space. It does not include HISTORY spaces with their related auxiliary objects when a table space with system-maintained temporal tables is specified in SYSIN. You may include HISTORY objects with the HISTORY keyword. See “HISTORY” on page 130.

**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 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.
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.

**Note**
Specifying AUX … on the OPTIONS statement in SYSIN applies AUX … to all RECOVER statements. To override AUX in the OPTIONS statement, specify AUX … only on the RECOVER statements you wish to override.

**HISTORY**

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

AUX must be specified to use the HISTORY keyword. The meaning of the HISTORY keyword depends on the AUX option specified as follows:

- **AUX NO HISTORY** includes regular tables, system-time base tables, and their related HISTORY tables. It does not include related XML or LOB objects.

- **AUX ALL HISTORY** includes regular tables, system-time base tables and their related HISTORY tables, and all related XML or LOB objects.

- **AUX XML HISTORY** includes regular tables, system-time base tables and their related HISTORY tables, and all related XML objects.

- **AUX LOB HISTORY** includes regular tables, system-time base tables and their related HISTORY tables, and all related LOB objects.

**USEHDROBIDS**

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

When you specify OBIDXLAT (“OBIDXLAT specification” on page 205) 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 ("USEHDROBIDS=YES" on page 610), 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.

**DEFINE**

DEFINE specifies if DEFINE NO objects will be instantiated.

---

**Note**

DB2 Version 11 or later supports the DEFINE option.

---

**DEFINE YES**

RECOVER PLUS instantiates DEFINE NO objects if necessary, and updates the catalog and DBD to reflect that the object has been defined.

**DEFINE NO**

RECOVER PLUS does not process DEFINE NO objects. NO is the default value.

**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. LOGSORT is required to properly order the log records.
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 ("BACKOUT=AUTO" on page 609), 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 481.

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 481.

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 (“ZIIP” on page 137 and “ZIIP=ENABLED” on page 601) 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 600).

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 (“ZIIP=ENABLED” on page 601).

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 (“XBMID=ssid or xbmGroup ” on page 600) or XBMID option on the OPTIONS command (“XBMID ssid or xbmGroup ” on page 136).

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 options module for all REBUILD INDEX or RECOVER UNLOADKEYS, or RECOVERBUILDINDEX 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, or RECOVERBUILDINDEX command statements.

When performing a REBUILD INDEX or a RECOVER UNLOADKEYS or RECOVERBUILDINDEX 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 $xxxWKnn DD statements in the JCL
- specify only SORTDEVT and let BMCSORT determine the number of data sets

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

SORTDEV 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 (“SORTDYN” on page 124), 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 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.

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.

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.

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 325.

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.

**SIMULATE YES**

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 \textit{integer} variable can be any integer value between 0 and 2,147,483,646.

ON ERROR CONTINUE \textit{integer} allows \textit{integer} + 1 errors before RECOVER PLUS terminates. If \textit{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 50 or the Recovery Management for DB2 User Guide.

**MAXKSORT \textit{integer}**

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

\textit{Note}\hspace{1cm} 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:

\texttt{minimum\left(\frac{2 \times \text{the number of CPUs}}{12}\right)}

\textit{Note}\hspace{1cm} 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 597.

\textit{Note}\hspace{1cm} 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 235.

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 (“MAXLSORT integer ” on page 145 and “KSORTSHARE=YES” on page 607)

  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 is a number between 1 and the value specified for MAXKSORT.

- SxxxWK

  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 “Set the MAXKSORT option” on page 495.

**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 (“MAXLSORT=0” on page 606). 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:

- **LOGOU**nnn: 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.

- **LxxxWK**nnn: 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:

  ```
  L001WK01 DD...
  L001WK02 DD...
  L002WK01 DD...
  L002WK02 DD...
  L003WK01 DD...
  L003WK02 DD...
  ```

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 prints the output for each phase for each object in an AFRPRnnn data set (“DD statements common to all RECOVER PLUS executions” on page 326). RECOVER PLUS names the files AFRPR001, AFRPR002, AFRPR003, and so on. The maximum number of AFRPRnnn files is calculated by the following formula:

\[ \text{value of MAXKSORT} + \text{value MAXLSORT} + 1 = \text{maximum number of AFRPRnnn files} \]

AFRPRINT contains a list of the phases executed for each object and the name of the AFRPRnnn data set where RECOVER PLUS put the output for the phase.

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 497.

**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 (“KSORTSHARE=YES” on page 607), 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 497.
AUTOSIZE

This option turns dynamic sizing for output image copies or change accumulation output files on or off. Valid values are YES or 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 output image copies or change accumulation output files to DASD occurs.

Note
Use AUTOSIZE=YES option when you want to use dynamic space calculation of OUTCOPY files generated by RECOVER PLUS. For more information, see “Automatic sizing of output copies dynamically allocated on DASD” on page 514.

AUTOSIZE NO

AUTOSIZE NO specifies that output image copies or change accumulation output files are allocated to DASD 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
  DSNAMES dataSetName
  RECOVER TABLESPACE databaseName.tableSpaceName
  TABLESPACE databaseName2.tableSpaceName2

OPTION RECOVERYPOINT URIDDDN(output) LOGPOINT X'logPoint'
  OUTPUT(output) UNIT SYSDA
  DSNAMES dataSetName
  RECOVER TABLESPACE databaseName.tableSpaceName
  TABLESPACE databaseName2.tableSpaceName2
```

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

- RECOVERYPOINT is not valid with the use of the ACCUM or IMPORT commands 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 in-flight 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 in-flight 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.D120206.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.

**OPTION RECOVERYPOINT TIMESTAMP 2013-01-15-09.15.00**

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 “TZRULE=NONE” on page 602.

**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 (“TORBA or TOLOGPOINT” on page 193) 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.
**SNAP**

The SNAP option indicates if you want RECOVER PLUS to read VSAM copies, even if the data set is not on a snappable disk.

If you do not specify SNAP on the OPTIONS command, RECOVER PLUS uses the value of the SNAP installation option ("SNAP=HW" on page 612) as the default value. The default value of the SNAP installation option is HW.

**SNAP HW**

When you specify SNAP HW, RECOVER PLUS uses a hardware snapshot to restore an Instant Snapshot or VSAM data set.

**SNAP VSAM**

When you specify SNAP VSAM, RECOVER PLUS uses conventional VSAM I/O to restore a VSAM data set if it is not on a snappable disk.

To read a VSAM copy with SNAP VSAM, you specify the name of the VSAM data set in an INCOPY statement, just as you would if it was an Instant Snapshot copy, using the INCOPY FULL SNAPSHOT DSNAME dataSetName syntax.

SNAP VSAM also allows you to recover using a VSAM copy registered in BMCXCOPY or SYSCOPY if that copy is not on snappable disk.

**OUTPUT**

This section describes the syntax of the OUTPUT command and its options.
OUTPUT syntax

The following figure shows the syntax of the OUTPUT command.

**Figure 9: OUTPUT syntax**

```
// OUTPUT  name  DSNNAME dataSetName
```

**Figure 10: OUTPUT syntax—Common options**

Common options

```
// UNIT  SYSALDA name
// CATLG  YES NO
// VOLCNT integer
// UNITCNT integer
// DATACLASS name
// MGMTCLASS name
// STORCLASS name
```

**Figure 11: OUTPUT syntax—Disk options**

Disk options

```
// SPACE (primary,secondary)
// CYL TRK
// VOLUMES (volume1,volume2,...)
// EATTR NO OPT NONE
// MAXPRIM integer
```

**Figure 12: OUTPUT syntax—Tape options**

Tape options

```
// STACK NO YES
// TRTCH
// COMP NOCOMP NONE
// EXPDT 999999 date
// RETPD integer
```
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 507.

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 (“OUTCOPYDDN specification” on page 220), or the RECOVERYDSN keyword (“RECOVERYDDN specification” on page 220).
You can construct dataSetName by using the symbolic variables in “Using symbolic variables” on page 510.

Table 6: Symbolic variables used with DSNAME and their descriptions

<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;STEPNAM</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>
<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 YYDDD)</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>Symbolic variable</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>&amp;TYPE</td>
<td>type of output being produced:</td>
</tr>
<tr>
<td></td>
<td>■ LP for local site primary</td>
</tr>
<tr>
<td></td>
<td>■ LB for local site backup</td>
</tr>
<tr>
<td></td>
<td>■ RP for recovery site primary</td>
</tr>
<tr>
<td></td>
<td>■ RB for recovery site backup</td>
</tr>
<tr>
<td>&amp;INST</td>
<td>instance number, with valid values of 1 or 2</td>
</tr>
<tr>
<td>&amp;UNIQ or &amp;UQ</td>
<td>1- to 8-character value, based on the system clock, that is used to generate unique copy data set names</td>
</tr>
<tr>
<td></td>
<td>The first character is always an uppercase letter. Each remaining character is either an uppercase letter or a numeral from 0 through 9.</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.

**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 any of the symbolic variables in “Using symbolic variables” on page 510.

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.

---

**Note**

You can use RECOVER PLUS to automatically size the OUTCOPY files generated by RECOVER PLUS. For more information, see “Automatic sizing of output copies dynamically allocated on DASD” on page 514.
**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 “VOLCNT integer” on page 156). 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 ("EATTR=" on page 611). The EATTR installation option defaults to EATTR=, which is the equivalent of EATTR=None.

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.

You can also set EATTR to OPT or NO in the JCL.

When you create an OUTCOPY (“OUTCOPY specification” on page 212) and want to force the copy to the cylinder-managed space of an EAV, specify the SPACE keyword (“INCOPY specification” on page 217).

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.

**MAXPRIM integer**

Specifying MAXPRIM allows you to set a maximum amount of disk space (in the units specified by SPACE) to allocate as primary space. Valid values are 0 through 65535. A nonzero value for integer establishes an upper limit for primary space allocation, while a value of zero specifies no limit.

*Note*

Although this option is not required for use with the AUTOSIZE=YES option (“AUTOSIZE” on page 147), you might find it useful when you want to use dynamic space calculation of OUTCOPY files generated by RECOVER PLUS. For more information, see “Automatic sizing of output copies dynamically allocated on DASD” on page 514.

**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 510.

**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).
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.

RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE

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

The following figures show the syntax of the RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE commands.
Figure 13: RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax diagram
Figure 14: RECOVER syntax—Table space specification

Table space specification

TABLESPACE  
  tableSpaceName  
    DSNDB04,  
      databaseName.  
    OBJECTSET specification  
      INDEXES  
        INDEX  
          NO  
          YES  
    INDEXES  
      INDEX  
        NO  
        YES  
      HISTORY  
    DROPRECOVERY  
      NO  
      YES  
    RICHK  
      NO  
      YES  
    DSNUM  
      ALL  
        Integer  
          begin : end  
    AUX  
      NO  
        ALL  
        XML  
        LOB  
    OUTCOPY specification  
      specification  
    TRANSFORM  
      specification  
    FROMRBA or FROMLOGPOINT specification  
      specification  
    Non-registered copy or INLOG specification  
      specification

1 Not valid with TRANSFORM
2 Not valid with BACKOUT
Figure 15: RECOVER syntax—Single index specification

```plaintext
Single index specification

INDEX (indexName)
  creatorID.
  OBJECTSET specification

INDEXSPACE (databaseName, indexSpaceName)
  DSNDB04.

DSNUM ALL
  integer
  begin : end

NOWORKDDN

OBIDXLAT specification

TRANSFORM

Non-registered copy or INLOG specification

FROMRBA or FROMLOGPOINT specification

OUTCOPY specification

¹ Valid with INDEXLOG AUTO.
Figure 16: RECOVER syntax—Multiple index specification

Multiple index specification

INDEX

(- ALL -) TABLESPACE

INDEXSPACE

databaseName

INDEXSPACE

(- ALL -) TABLESPACE

NOWORKDDN²

OUTCOPY specification¹

¹ Allowed with multiple recoveries only if dynamic outcopy allocation (OUTPUT) syntax is used.
² Valid with INDEXLOG AUTO.
Figure 17: RECOVER syntax—OBJECTSET specification

OBJECTSET specification

OBJECTSET  creator.name

EXCLUDE

(dbname.ts qualifier.ix)

DSNUM

ALL

integer

begin : end

OBIDXLAT

RESET

Figure 18: RECOVER syntax—INDEPENDENT OUTSPACE specification

INDEPENDENT OUTSPACE specification

INDEP OUTSPACE

MODEL

vcat.BMCDBC.databaseName.objectName.10001.znnm

userNamedDataSet

\(^1\) Not valid with BACKOUT
**Figure 19: RECOVER syntax—TORBA or TOLOGPOINT specification**

**TORBA or TOLOGPOINT specification**

```
TORBA
TOLOGPOINT

LASTQUIESCE

LASTCOMMONQ
LASTARCHQ
LASTSHUTDOWN

LOGMARK logMarkName

X'logPoint'

(relativeGenerationNumber)

BACKOUT
YES
```

**Figure 20: RECOVER syntax—TOCOPY specification**

**TOCOPY specification**

```
TOCOPY

LASTCOPY

datasetName

(relativeGenerationNumber)

TOVOLUME

CATALOG

volumeSerialNumber

TOSEQNO integer

* Not valid with OBJECTSET
```
Figure 21: RECOVER syntax—LOGSORT specification

```
LOGSORT specification

- LOGSORT
  - SORTDEVT deviceType
  - SORTNUM integer
  - NUMREC
    - CALC
    - NOEST
    - EST integer
      - AVGRECSZ integer
    - ABS integer
      - AVGRECSZ integer
```

Figure 22: RECOVER syntax—OBIDXLAT specification

OBIDXLAT specification

1 Not valid with BACKOUT
Figure 23: RECOVER syntax—Non-registered copy or INLOG specification

**Non-registered copy or INLOG specification**

- **INCOPY** — FULL
  - **INLINE**
    - **SNAPSHOT**
  - **INLOG RBA**
  - **INLOG LOGPOINT**
  - **X'logPoint'**

1 Not valid with index specification
2 Not valid with BACKOUT

Figure 24: RECOVER syntax—FROMRBA or FROMLOGPOINT specification

**FROMRBA or FROMLOGPOINT specification**

- **FROMRBA**
- **FROMLOGPOINT**
  - **LASTQUIESCE**
  - **LASTCOPY**
  - **LASTARCHQ**
  - **LASTSHUTDOWN**
  - **X'logPoint'**

Figure 25: RECOVER syntax—OUTCOPY specification

**OUTCOPY specification**

- **OUTCOPY**
  - **NO**
  - **YES**
    - **ONLY**
      - **REGISTER**
        - **ALL**
        - **NONE**
  - **ddname**

1 Not valid with BACKOUT or Instant Snapshot copies
Figure 26: RECOVER syntax—OUTCOPY specification, continued
Figure 27: RECOVER syntax—INCOPY specification

INCOPY specification

DSNAME dataSetName

MODEL dataSetName

DSSIZE (integerG)¹

PIECESIZE (...¹

integerK

integerM

integerG

RBA

LOGPOINT

X'logPoint'

SHRLEVEL

REFERENCE

CHANGE

ENCrypted

timestamp

COMPRESSED ²

INVOICE

CATALOG

volumeSerialNumber

INDEV DeviceType

INSEQNO integer

¹ Valid only with TRANSFORM
² Not valid with table space specification

RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE
Figure 28: RECOVER syntax—OUTCOPYDDN specification

OUTCOPYDDN specification

```
OUTCOPYDDN ( DDName1, DDNamePrefix1, outputDescriptor1, DDName2, DDNamePrefix2, outputDescriptor2 )
```

1 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.
2 These values are the default values used by the OUTPUT command.
3 These values are used to specify the OUTPUT command, which will control the allocation of the copy.

Figure 29: RECOVER syntax—RECOVERYDDN specification

RECOVERYDDN specification

```
RECOVERYDDN ( DDName3, DDNamePrefix3, outputDescriptor3 )
```

```
RECOVERYDSN ( dataSetName3, dataSetName4 )
```

1 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.
2 These values are the default values used by the OUTPUT command.
3 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

The following table describes the common dependencies for coding the RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX commands.

Table 7: RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX command dependencies

<table>
<thead>
<tr>
<th>Option coded</th>
<th>Option that must be used</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCOPY</td>
<td>TOCOPY LASTCOPY&lt;br&gt;OR&lt;br&gt;TOCOPY data set name where data set name matches the last DSNAME in INCOPY&lt;br&gt;OR&lt;br&gt;RBA/LOGPOINT on the last INCOPY specification</td>
</tr>
<tr>
<td>OBIDXLAT and INCOPY</td>
<td>TOCOPY on the RECOVER TABLESPACE command statements&lt;br&gt;OR&lt;br&gt;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>
<tr>
<td>INCOPY and OBJECTSET</td>
<td>MODEL with symbolics</td>
</tr>
<tr>
<td>INDEX(ALL) and INDEP OUTSPACE</td>
<td>MODEL with symbolics</td>
</tr>
</tbody>
</table>

Table 8 on page 174 describes the common prohibitions to coding the RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX commands.

Table 8: RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX command prohibitions

<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&lt;br&gt;OR&lt;br&gt;LOGONLY</td>
</tr>
<tr>
<td>INCOPY or INLOG</td>
<td>LOGAPPLY ONLY&lt;br&gt;OR&lt;br&gt;LOGONLY</td>
</tr>
<tr>
<td>Option coded</td>
<td>Option that must not be used</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>------------------------------------------------------------------</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>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 “RECOVER TABLESPACE, RECOVER INDEXSPACE, and RECOVER INDEX command dependencies and prohibitions” on page 174 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 151.) 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 or 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 or 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(\textit{x},\textit{y}) clause, where \textit{x} is the ID of the source table and \textit{y} is the ID of the target table. If you do not specify this extra OBID clause, the job issues the following message:

\begin{quote}
BMC40764I TABLE OBID X'xxxx' FOUND AT OFFSET X'3A' IN INDEX HEADER WILL NOT BE TRANSLATED
\end{quote}

You may, however, omit both OBID(\textit{x},\textit{y}) clauses. If you omit both clauses, appropriate default values are supplied. For more information, see “OBIDXLAT specification” on page 205.
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 141.

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**

For option descriptions for specifications which are included in the Table space specification, see

- “OBIDXLAT specification” on page 205
- “Non-registered copy or INLOG specification” on page 207
- “FROMRBA or FROMLOGPOINT specification” on page 210
- “OUTCOPY specification” on page 212

The syntax diagram for the Table Space specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

**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 specification**

Use the OBJECTSET specification with TABLESPACE to recover only the table spaces that are in a defined group.

For more information, see “OBJECTSET specification” on page 187 and “Using BMC RECOVERY MANAGER groups” on page 533.

**Note**

To include the indexes in a recovery using TABLESPACE OBJECTSET, specify INDEXES YES.

**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 128.

RICHK

RICHK specifies if RECOVER PLUS will check for referential integrity constraints on the objects being recovered.

RICHK YES

RICHK YES checks for referential integrity constraints and sets check pending (CHKP) status for dependent objects not recovered in the same step to the same PIT.

RICHK NO

RICHK NO does not check for referential integrity. NO is the default value.

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 or 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 or TOLOGPOINT option. If you do not specify a TORBA or 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 “Recovering a dropped table space or table” on page 539.

**TRANSFORM**

Use TRANSFORM to specify that you want to do a DDL transformation. A TOCOPY LASTCOPY and OBIID 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) table space
- non-large partitioned table space to large partitioned table space
- range-partitioned table space to range-partitioned (PBR) universal table space
- range-partitioned table space to partition-by-growth universal table space
- multi-table table space to partition-by-growth universal table space

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:

- change the segment size (SEGSIZE) of a table space
- transform a non-segmented table space into a segmented one
- change the data set size (DSSIZE or PIECESIZE) of your table space or index
- change the compression attribute of your index
- add or remove the MEMBER CLUSTER attribute
Multi-table table space transformations have the following requirements:

- Multi-table table spaces must be segmented.
- Each table must be processed with a separate TRANSFORM statement.
- The TRANSFORM statement must include OBIDXLAT OBID with a source OBID value. The target OBID can be defaulted. The DBID and PSID are not required.
- The target indexes must be rebuilt.

**Note**
The transformation requires the RBA/LRSN format of the source and target objects to match.

Figure 30 on page 182 provides an example of the syntax used to transform a multi-table table space into two separate single-table table spaces.

**Figure 30: Example syntax for a multi-table table space transformation**

```sql
RECOVER TABLESPACE PBGDB.TS1 DSNUM ALL
   OBIDXLAT OBID(3,3)
   TRANSFORM
      INCOPY FULL SNAPSHOT
      MODEL RDADMB.DSNDBD.MULTDB.MULTTS.SNAP&DSNUM
   REBUILD INDEX(ALL) TABLESPACE PBGDB.TS1 DSNUM ALL

RECOVER TABLESPACE PBGDB.TS2 DSNUM ALL
   OBIDXLAT OBID(7,3)
   TRANSFORM
      INCOPY FULL SNAPSHOT
      MODEL RDADMB.DSNDBD.MULTDB.MULTTS.SNAP&DSNUM
   REBUILD INDEX(ALL) TABLESPACE PBGDB.TS2 DSNUM ALL
```

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

**Single index specification**

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 205
- “Non-registered copy or INLOG specification” on page 207
- “FROMRBA or FROMLOGPOINT specification” on page 210
- “OUTCOPY specification” on page 212
The syntax diagram for the Single index specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

**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 specification`

Use the `OBJECTSET` specification with `INDEX` to recover only the indexes that are in a defined group. For more information, see “OBJECTSET specification” on page 187 and “Using BMC RECOVERY MANAGER groups” on page 533.

`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 (“PART” on page 234) 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 DSNICOPY, 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

For option descriptions for the OUTCOPY specification, which is included in the Multiple index specification, see

- “OUTCOPY specification” on page 212

The syntax diagram for the Multiple index specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

**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 specification**

Use the OBJECTSET specification with INDEX to recover only the indexes that are in a defined group. For more information, see “OBJECTSET specification” on page 187 and “Using BMC RECOVERY MANAGER groups” on page 533.

**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 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 recovery of the base table spaces. For more information, see “AUX” on page 128.

**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.

The OBJECTSET specification is shown in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

OBJECTSET creator.name

OBJECTSET is available in the following syntactical forms:

- RECOVER OBJECTSET creator.name
  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
  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
  RECOVER INDEX 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* 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 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 can specify OBIDXLAT with OBJECTSET only when all the OBID specifications can be omitted. See OBID specification on page “OBIDXLAT” on page 206 for further information.

- Only TOCOPY LASTCOPY is valid with OBJECTSET. (You cannot use a specific data set name.)

- You cannot use INLOG with OBJECTSET.

- RECOVER PLUS allows only one OBJECTSET clause, and the statement cannot 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.

**EXCLUDE (dbname.ts or qualifier.ix )**

Use the EXCLUDE option after an OBJECTSET specification to exclude one or more objects from recovery. You can use wildcards or fully qualified names to specify the exclusions. The following wildcards are valid:

- % and * match no characters to any number of characters in the object name.

- _ and ? match a single character in the object name.
List the excluded objects following the EXCLUDE keyword. Each item in the list must be in the form `dbname.ts` or `qualifier.ix` and you must separate the individual items with commas. Enclose the list in parentheses.

The following considerations apply to use of the EXCLUDE option with the OBJECTSET specification:

- When you specify INDEXES YES for a RECOVER OBJECTSET statement and specify a tablespace for exclusion, the associated indexes are also excluded.
- You cannot exclude individual indexes when INDEXES YES is specified.

**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.

**DSNUM integer**

Use this option to exclude 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 exclude 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.

**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, 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.

INDEPENDENT OUTSPACE specification

The syntax diagram for the INDEPENDENT OUTSPACE specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

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
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.

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.

- 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 85 and “Secondary extents” on page 85.

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 A00 n 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 “STOGROUP specification description” on page 123). 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 DSNDBC 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 “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161. Also, see “Testing recovery with simulation” on page 55 and “Examples of RECOVER PLUS jobs” on page 377 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 the symbolic variables “Using symbolic variables” on page 510.
TORBA or TOLOGPOINT specification

The syntax diagram for the TORBA or TOLOGPOINT specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

Note
Place syntax for the TORBA or TOLOGPOINT specification after the syntax for Table space specification, Single index specification, or Multiple index specification.

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 or TOLOGPOINT with table spaces and indexes, see “RECOVER INDEX and RECOVER INDEXSPACE command dependencies and prohibitions” on page 175.)

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.

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 (< = 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 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.

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:
INDEPENDENT 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

TOCOPY specifies the last image copy to be processed.

The syntax diagram for the TOCOPY specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

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 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) 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.

Note

TOCOPY dataSetName is not allowed with OBJECTSET.

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 119. The syntax diagram for the LOGSORT specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

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 HPGRRBRBA 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 installation options module for local site:
**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 installation options module for remote site:

<table>
<thead>
<tr>
<th>DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMAGE COPIES</td>
</tr>
<tr>
<td>(RP,RB)</td>
</tr>
<tr>
<td>(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.

**DEFINE**

DEFINE specifies if DEFINE NO objects will be instantiated.

---

*Note*

DB2 Version 11 or later supports the DEFINE option.

**DEFINE YES**

RECOVER PLUS instantiates DEFINE NO objects if necessary, and updates the catalog and DBD to reflect that the object has been defined.

**DEFINE NO**

RECOVER PLUS does not process DEFINE NO objects. NO is the default value.

**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 325.

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 84.

**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.

  **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 190. 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 input copy has system pages for all versions of the data. One way to do this is to use COPY PLUS with the GENSYSYPAGES option to make the copy.
- recover table spaces in a separate step from rebuilding indexes

**OBIDXLAT specification**

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 is specified without DROPRECOVERY. If INCOPY 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 is specified without DROPRECOVERY. If INCOPY 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 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 (“USEHDROBIDS” on page 130 and “USEHDROBIDS=YES” on page 610) 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

```
RECOVER ... OBIDXLAT DBID(,101) PSID(,201)
INCOPY FULL DSNAME ...
TOCOPY LASTCOPY
```

The following example is *not* valid:

```
RECOVER ... OBIDXLAT DBID(,101) PSID(,201)
INCOPY FULL DSNAME ... RBA X'123456'
```

The syntax diagram for the OBIDXLAT specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

**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, 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 \((X'\text{hexSourceID}', X'\text{hexTargetID}')\)

DBID \((\text{decimalSourceID}, \text{decimalTargetID})\)

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 205.

PSID \((X'\text{hexSourceID}', X'\text{hexTargetID}')\)

PSID \((\text{decimalSourceID}, \text{decimalTargetID})\)

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 205.

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 \((X'\text{hexSourceID}', X'\text{hexTargetID}')\)

OBID \((\text{decimalSourceID}, \text{decimalTargetID})\)

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 205.

**Non-registered copy or INLOG specification**

The syntax diagram for the Non-Registered Copy or INLOG Use specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.
**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 “TOCOPY specification” on page 197). 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 (“INCOPY specification” on page 217) 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 484.)

FLASHCOPY

FLASHCOPY specifies that the copy specified in the INCOPY specification is an IBM FlashCopy. 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 217) 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

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.

The syntax diagram for the FROMRBA or FROMLOGPOINT specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.
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

For option descriptions for the OUTCOPYDDN and RECOVERYDDN specifications, which are included in the OUTCOPY specification, see

- “OUTCOPY specification” on page 212
- “RECOVERYDDN specification” on page 221

The syntax diagram for the OUTCOPY specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

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 220 and “RECOVERYDDN specification” on page 221. The copies that you want to register are specified by the REGISTER option (“INCOPY specification” on page 217).

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.

Note

You can use RECOVER PLUS to automatically size the OUTCOPY files generated by RECOVER PLUS. For more information, see “Automatic sizing of output copies dynamically allocated on DASD” on page 514.
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 127.

- 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 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.SYSCOPY 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 “OUTCOPY specification” on page 212, “RECOVERYDDN specification” on page 221, and “Copy data set ddname construction for JCL-allocated data sets” on page 222.
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.

**TAG tagName**

Specify TAG to tag copies with a name that can be used for selecting objects for use with the COPY PLUS EXPORT command. Only image copies for table spaces are tagged.

Tag names (tagName) are presented as 1 to 32 characters, followed by a generation number in parentheses. The generation number can be (0), (+1), or (-n). Generation numbers are used as follows:
- (0)—uses the highest generation of the tag.
- (+1)—uses the highest generation of the tag, plus 1.
- (-n)—uses the highest generation of the tag, less n generations.

Spaces are not allowed in the tag name. Specify the generation number as in the following example: WEEKLY_EXPORT(+1).

**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 “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

**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* allows you to construct the copy data set by using symbolic variables. This is useful when used with OBJECTSET processing.
**Note**
When you use INCOPY with TRANSFORM, specify the VSAM data component instead of the VSAM cluster. `dataSetName` must include a node for the data set number represented by `A&LDSNUM`. This node can appear anywhere within `dataSetName`.

You can construct `dataSetName` by using the symbolic variables in “Using symbolic variables” on page 510.

**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, 64G, 128G, and 256G. 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 521.

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 "KEYDSNAM= keyDataSetName " on page 605) RECOVER PLUS uses to decode the copy. The timestamp value uses either of the following formats:

\[ yyyy-mm-dd-hh.mm.ss \]

\[ yyyy-mm-dd-hh.mm.ss.nnnnnn \]

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 “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.
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 222.

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 “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.
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” on page 222.

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 namennn, 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 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 205.
**REBUILD INDEX syntax**

The following figures show the syntax of the REBUILD INDEX command.

**Figure 31: REBUILD INDEX syntax diagram**

---

1 Is synonymous to REBUILD when INDEXLOG=NO
2 Is synonymous to SIMRBLD when INDEXLOG=NO
Figure 32: REBUILD INDEX syntax diagram, continued

Figure 33: REBUILD INDEX syntax diagram—OBJECTSET specification

**OBJECTSET specification**

```
OBJECTSET (creator.name)

EXCLUDE (dbname.ts, qualifier.ix)

DSNUM (ALL, integer, begin : end)
```
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 177.

**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 141.

**SIMRCVR**

SIMRCVR is a synonym for SIMRBLD when INDEXLOG is set to NO. More move information, see “SIMRBLD” on page 227.

**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 specification**

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 OBJECTSET specification is shown in “REBUILD INDEX syntax” on page 225.

**OBJECTSET creator.name**

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 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 INTABLESPACE with OBJECTSET.
- You cannot use TABLESPACE with OBJECTSET.

**EXCLUDE (dbname.ts or qualifier.ix)**

Use the EXCLUDE option after an OBJECTSET specification to exclude one or more objects from the rebuild. You can use wildcards or specific names to specify the exclusions. The following wildcards are valid:

- `%` and `*` match any number of characters in the object name.
- `_` and `?` match a single character in the object name.

List the excluded objects following the EXCLUDE keyword. Each item in the list must be in the form `dbname.ts` or `qualifier.ix` and you must separate the individual items with commas. Enclose the list in parentheses.

**DSNUM**

DSNUM specifies a single data set in the specified space or the entire space for rebuild.

**DSNUM ALL**

DSNUM ALL is the default and specifies that all of the data sets in the space are to be rebuilt. If the space is nonpartitioned and DSNUM ALL is specified or implied, image copies made for specific data sets will only be considered for rebuild if they are registered at the same log point.

**DSNUM integer**

Use this option to rebuild only one partition or data set in the space. For partitioned spaces, DSNUM `integer` specifies the number (from 1 through 4096) of a single partition in the specified space. For nonpartitioned spaces, DSNUM `integer` specifies the number (from 1 to 32) of a single data set of the specified space.

**DSNUM begin : end**

Use this option to rebuild a range of partitions in the space. 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.

**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 DSNDDB01, DSNDDB06, DSNDDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE. The default is DSNDDB04.

- 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 128.

**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.

- 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 85 and “Secondary extents” on page 85.

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 “STOGROUP specification description” on page 123).

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 “REBUILD INDEX syntax” on page 225.

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 417. For more information about this feature, see “Testing recovery with simulation” on page 55.

**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 “Stacking copies on tape” on page 510.

**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 “`REBUILD INDEX syntax`” on page 225.

For information about using redirection with multiple `RECOVER PLUS` commands, see “Example 9: Recovering to a non-DB2 data set” on page 417.

**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 but may change to UNLOADnn for table spaces with fewer than 100 partitions when REBUILD INDEX is accompanied by one or more RECOVER UNLOADKEYS statements. For table spaces with 100 or more partitions, the default value is UNLOADnnn. WORKDDN is required when REBUILD INDEX is
accompanied by RECOVERUNLOADKEYS for the same partition and NOUKSORT is specified.

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 “MAXKSORT integer” on page 143 and “MAXKSORT” on page 597.

If you specify RECOVERUNLOADKEYS and NOUKSORT together with REBUILD INDEX for the same table space and partitions, the WORKDDN ddname on both statements must be the same. The ddname is considered a prefix and must not be more than six characters for spaces with fewer than 100 partitions or not more than five characters for spaces with 100 or more partitions.

For more information, see “Restrictions for key work data sets, SKEYDDN, SORTNUM, and SORTDEVT” on page 95.

**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 ("SORTDYN" on page 124), 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.

**DEFINE**

DEFINE specifies if DEFINE NO objects will be instantiated.

*Note*

DB2 Version 11 or later supports the DEFINE option.

**DEFINE YES**

RECOVER PLUS instantiates DEFINE NO objects if necessary, and updates the catalog and DBD to reflect that the object has been defined.

**DEFINE NO**

RECOVER PLUS does not process DEFINE NO objects. NO is the default value.
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 “RECOVER PLUS concepts” on page 467.

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 84.

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 231.

---

**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.

**SHRLEVEL**

The SHRLEVEL option specifies the level of access that concurrently operating DB2 applications and utilities should have to the target table space or index space. You can specify read-only access or read-write access. If you do not specify this option, RECOVER PLUS allows read-only access to the target table space during the rebuild process.

**SHRLEVEL REFERENCE**

SHRLEVEL REFERENCE is the default and allows read-only access by other programs to the target space during the rebuild process.
SHRLEVEL CHANGE

SHRLEVEL CHANGE allows read-write application access to DB2 table space and index space objects during the rebuild process.

Performing an online rebuild index by specifying SHRLEVEL CHANGE offers the following benefits:

- Allows full access to DB2 data during most of the rebuild
- Delivers significantly improved data availability by greatly reducing the outage for the DB2 objects
- Operates in a nondestructive manner, which allows you to make the objects available without having to recover in the event of a failure

To use the SHRLEVEL CHANGE option, you must have installed the BMC EXTENDED BUFFER MANAGER (XBM) product or SNAPSHOT UPGRADE FEATURE (SUF) component of XBM.

**Note**

SHRLEVEL CHANGE is not allowed for the indexes in RBDP or PSRBD status and VCAT-defined table spaces.

MAXRO

MAXRO specifies the maximum length of time for the last iteration of log processing.

*integer*

This integer specifies the maximum number of seconds that RECOVER PLUS spends applying log records for the last iteration of log processing. Specify this value as a positive integer (0 or greater). During that iteration, applications have no access. The accrual execution time of the last iteration might exceed the specified value for MAXRO.

Specifying a small positive value minimizes the period of no access, but it might increase the elapsed time for REBUILD INDEX to complete. If you specify a very large positive value, the second iteration of log processing will probably be the last iteration.

The default value is the value of the lock timeout system parameter IRLMRWT.
DEFER

This value tells RECOVER PLUS to continue applying log records indefinitely, regardless of the value set in any other SHRLEVEL CHANGE option.

LONGLOG

LONGLOG specifies the action to take if RECOVER PLUS determines that a longlog condition exists. A longlog condition exists when RECOVER PLUS determines over a period of time that the DB2 subsystem is generating log records for the objects that you are rebuilding faster than RECOVER PLUS is applying the records. After detecting that a longlog condition exists, RECOVER PLUS sends an action write-to-operator (WTO) notification and continues to apply the log records until the delay interval specified on the DELAY option expires. If the longlog condition still exists, RECOVER PLUS takes the action that you specify with one of the following options.

CONTINUE

CONTINUE indicates that RECOVER PLUS is to continue processing.

TERM

TERM indicates that RECOVER PLUS is to terminate the reorganization as specified by DELAY.

DRAIN

DRAIN indicates that RECOVER PLUS is to drain the write claim class specified by DELAY, which applies the remaining log records.

DELAY

DELAY specifies the number of seconds that are to elapse from the time RECOVER PLUS detects a longlog condition until it performs the action specified by the LONGLOG option. If the longlog condition no longer exists at the end of the time period specified by DELAY, the timer is reset. When RECOVER PLUS detects the next longlog condition, it restarts the timer, using the original DELAY value. The value must be a positive integer (0 or greater). The default value is 1200.

DRAIN_WAIT

DRAIN_WAIT specifies the number of seconds that REBUILD INDEX is to wait when draining the table space or index. The specified time is the aggregate time for objects that are to be checked.

integer can be any integer from 0 to 1800. If you do not specify DRAIN_WAIT or if you specify a value of 0, RECOVER PLUS uses the value of the lock timeout subsystem parameter IRLMRWT.
RETRY

RETRY specifies the maximum number of retries that REBUILD INDEX should attempt.

integer can be any integer from 0 to 255. REBUILD INDEX uses 1 as the default if you do not specify a value.

RETRY_DELAY

RETRY_DELAY specifies the minimum number of seconds between retries.

integer can be any integer from 1 to 1800.

If you do not specify RETRY_DELAY, REBUILD INDEX uses the DRAIN_WAIT value x RETRY value.

RBALRSN_CONVERSION

The RBALRSN_CONVERSION option specifies the RBA or LRSN format of the index.

If this option is not specified, UTILITY_OBJECT_CONVERSION from the DSNZPARM determines the RBA or LRSN format. RBALRSN_CONVERSION cannot be used with indexes on table spaces that are involved in a clone relationship.

NONE

NONE tells RECOVER PLUS to rebuild the index with the current format defined by the column RBA_FORMAT in SYSIBM.SYSINDEXES.

BASIC

BASIC tells RECOVER PLUS to convert the RBA or LRSN format to the 6-byte format if the original format is EXTENDED. The job will fail if UTILITY_OBJECT_CONVERSION is set to NOBASIC.

EXTENDED

EXTENDED tells RECOVER PLUS to convert the RBA or LRSN format to the 10-byte format if the original format is BASIC.

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

The following figure shows the syntax of the RECOVER UNLOADKEYS command.

**Figure 35: RECOVER UNLOADKEYS syntax diagram**

```
RECOVER UNLOADKEYS (indexName [PART 1 integer [begin : end] ]
| (ALL ) )

TABLESPACE tableSpaceName

INDEP INTABLESPACE MODEL vcat.BMCDB.C databaseName tableSpaceName l0001 znn

CLONE SKEYDDN SKEY DDName SORTDEV deviceType SORTNUM integer

NUMREC CALC NOEST EST integer ABS integer

ANALYZE YES NO ONLY
```

1 You cannot specify PART in both places.
RECOVER UNLOADKEYS option descriptions

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 141.

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 \textit{begin} : \textit{end} \\

Use this option to recover a range of partitions for the index space. For partitioned index spaces, \textit{begin} specifies the number (from 1 through 4095) of the first partition in the range and \textit{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.

\textbf{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.

\textbf{TABLESPACE} \textit{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(\textit{indexName1}, \ldots) 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.

\begin{itemize}
  \item \textit{databaseName} is the name of the database to which the table space belongs. The default is DSNDDB04. The name cannot be DSNDDB01, DSNDDB06, DSNDDB07, or any database defined with TYPE='W' in SYSIBM.SYSDATABASE
  \item \textit{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.
\end{itemize}

\textbf{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 (\textit{indexName1}, \textit{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.

\textbf{PART} \textit{integer} \\

PART \textit{integer} specifies the partition of the table space from which the keys and row IDs are extracted. \textit{You must always specify this option with RECOVER UNLOADKEYS.} \\

\textit{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) INTO TABLESPACE 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:

(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

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 “RECOVER UNLOADKEYS syntax” on page 246.

For information about using redirection with multiple RECOVER PLUS commands, see “Example 9: Recovering to a non-DB2 data set” on page 417. For more information about some applications of this feature, see “Testing recovery with simulation” on page 55.

**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.

**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 (sort work) 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 or SORTNUM.

**SORTDEVT deviceType**

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

If you specify SORDYN NO ("SORDYN" on page 124), 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, \textit{n} equals 32. If you specify \textit{integer} greater than 32, \textit{n} equals \textit{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 SORTNUM, RECOVER PLUS uses the installation option default.

\textbf{NUMREC}

You must specify the same NUMREC values for all RECOVER UNLOADKEYS command statements for the same table space. 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 “RECOVER PLUS concepts” on page 467.

\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 NUMREC or that you specify NUMREC CALC (the default).

\textbf{NUMREC CALC}

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 section 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

The following figure shows the syntax of the RECOVER BUILDINDEX command.

Figure 36: 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 141.

**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.

\[(indexName1,indexName2, \ldots \ldots)\]

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.

- 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 85 and “Secondary extents” on page 85.

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 A00 n 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 “STOGROUP specification description” on page 123).

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

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 “RECOVER BUILDINDEX syntax” on page 255.

For information about using redirection with multiple RECOVER PLUS commands, see “Example 9: Recovering to a non-DB2 data set” on page 417. For information about some applications of this feature, see “Testing recovery with simulation” on page 55.

**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 “Stacking copies on tape” on page 510.

**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 DSNDDB04. The name cannot be DSNDDB01, DSNDDB06, DSNDDB07, 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 84.

**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 256.

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.

LOGSCAN

This section describes the syntax of the LOGSCAN command and its options.

LOGSCAN syntax

The following figures show the syntax of the LOGSCAN command.

Figure 37: LOGSCAN syntax diagram

LOGSCAN

Objectset creator.name

Table space specification

Single index specification

Multiple index specification

TORBA or TOLOGPOINT recovery specification

LOGAPPONLY ¹

LOGONLY ¹

¹ Not valid with BACKOUT, which is part of the TORBA or TOLOGPOINT specification
Figure 38: LOGSCAN syntax—Table space specification

Table space specification

```
TABLESPACE
   tableSpaceName
      DSND04.
databaseName.
     creating
      INDEXES
         NO
            INDEX
                YES

      DSNM
        ALL
           integer
              begin : end
      AUX
        NO
           ALL
              XML
                  LOB

      FROMRBA
      FROMLOGPOINT

      LASTQUIESCE
      LASTCOPY
      LASTARCHQ
      LASTSHUTDWN
      X'logPoint'
```

Figure 39: LOGSCAN syntax—Single index specification

Single index specification

```
INDEX
   ( 
      objectID.
      indexName
   )

INDEXSPACE
   ( 
      databaseName.
      indexSpaceName
   )

      DSNM
        ALL
           integer
              begin : end

      FROMRBA
      FROMLOGPOINT

      LASTQUIESCE
      LASTCOPY
      LASTARCHQ
      LASTSHUTDWN
      X'logPoint'
```
Figure 40: LOGSCAN syntax—Multiple index specification

Multiple index specification

Figure 41: LOGSCAN syntax—TORBA or TOLOGPOINT specification

TORBA or TOLOGPOINT 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 BUILDDINDEX
  - REBUILD INDEX
  - ACCUM

- 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 or 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 or 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 or 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 “LOGSCAN syntax” on page 261.
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 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 process only the table spaces that are in a defined group. For more information, see “OBJECTSET specification” on page 272 and “Using BMC RECOVERY MANAGER groups” on page 533.

Note

To include the indexes in a log scan using TABLESPACE OBJECTSET, specify INDEXES YES.

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).
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 128.

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.

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 “LOGSCAN syntax” on page 261.

**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 272 and “Using BMC RECOVERY MANAGER groups” on page 533.

**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 “Table space specification” on page 265.

**Multiple index specification**

The syntax diagram for the Multiple index specification is in “LOGSCAN syntax” on page 261.

**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 272 and “Using BMC RECOVERY MANAGER groups” on page 533.

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:

<table>
<thead>
<tr>
<th>databaseName</th>
<th>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.</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableSpaceName</td>
<td>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.</td>
</tr>
</tbody>
</table>

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 128.

INDEXSPACE databaseName.indexSpaceName

The INDEXSPACE option specifies indexes to be scanned as follows:

<table>
<thead>
<tr>
<th>databaseName</th>
<th>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.</th>
</tr>
</thead>
<tbody>
<tr>
<td>indexSpaceName</td>
<td>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.</td>
</tr>
</tbody>
</table>

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.

The OBJECTSET specification is shown in “LOGSCAN syntax” on page 261

- **LOGSCAN OBJECTSET creator.name**
  
  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 TABLESPACE OBJECTSET creator.name**
  
  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*
  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.

**TORBA or TOLOGPOINT specification**
The following section describes the TORBA or TOLOGPOINT specification shown in “LOGSCAN syntax” on page 261.

**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 264.)

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 common 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.
- 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.

MIGRATE

The MIGRATE command allows you to move data from a one or more table spaces to another. The table spaces can be on the same subsystem, or they can be on different subsystems, as long as the two subsystems share DASD.

A migration file that the COPY PLUS EXPORT command creates facilitates the migration. You run the EXPORT command on the target subsystem to collect information on one or more target table spaces. This information includes object identifiers and other information relating to the target table spaces. You run the RECOVER PLUS MIGRATE command on the source subsystem to migrate all the
table spaces to the target table spaces that are identified by the command. The INDEP OUTSPACE specification is required in order to specify the target VSAM data set model name. Prior to executing the MIGRATE command, the target table spaces must be stopped.

The MIGRATE command queries the source SYSCOPY and BMCXCOPY tables for the required image copy, and then merges the copy with log (if requested) and translates object identifiers to create the target image. The log information can come from DB2 log and change accumulation files. If you specify FROMRBA, only log is used for the migration.

**Note**
This command requires one of the following valid passwords:
- a Recovery Management solution password
- a Database Administration solution password
The Database Administration (DAD) password allows RECOVER PLUS to run within a worklist, but does not include full-function RECOVER PLUS.

For more information about copy migration using COPY PLUS EXPORT and RECOVER PLUS MIGRATE, see “Using MIGRATE and IMPORT for data movement” on page 535 and the Recovery Management User Guide.

**MIGRATE syntax**

The following figures show the syntax of the MIGRATE command.
Figure 42: MIGRATE syntax

1 Requires a valid password as follows: Recovery Management or Database Administration
Figure 43: MIGRATE syntax—USING specification

**USING specification**

- **USING** *(dataSelName)*

  - **CHANGE TABLE** FROM *creator1.tableName1* TO *creator2.tableName2*
  - **CHANGE INDEX** FROM *creator1.indexName1* TO *creator2.indexName2*

- **INDEXES**
  - **RECOVER**
    - **NO**

- **EXCLUDE** *(dbname.ts, qualifier.ix)*

  - **DSNUM** *(integer, begin : end)*

  - **LASTQUIESCE**
  - **LASTCOPY**
  - **LASTARCHQ**
  - **LASTSHUTDOWN**
  - **X’logPoint’**

Figure 44: MIGRATE syntax—FROMRBA or FROMLOGPOINT specification
Figure 45: MIGRATE syntax—OBJECTSET specification

**OBJECTSET specification**

```
OBJECTSET  creator.name

EXCLUDE ( dbname.ts qualifier.ix )
```

Figure 46: MIGRATE syntax—INDEPENDENT OUTSPACE specification

**INDEPENDENT OUTSPACE specification**

```
INDEP OUTSPACE

MODEL vcätz.MCDBCh.databaseName.objectName.I0001.znnn

MODEL userNamedDataSt
```

Figure 47: MIGRATE syntax—TORBA or TOLOGPOINT specification

**TORBA or TOLOGPOINT specification**

```
TORBA
TOLOGPOINT

LASTQUIESCE (relativeGenerationNumber)

LASTCOMMONQ
LASTARCHQ
LASTSHUTDOWN
LOGMARK logMarkName (logMarkGeneration)
X'llogPoint'
```
The MIGRATE command has the following limitations:

- no support for copies of the DB2 catalog

- no support for specifying DSNUM

  When selecting image copies, the MIGRATE command uses a DSNUM 0 copy or DSNUM n copies. DSNUM n copies will only be considered when all the partitions or data sets are registered at the same log point.

- no support for rebuilding migrated indexes

  However, RECOVER PLUS can recover any indexes in the migration data set depending on the value of the INDEXES parameter in the USING clause. If the source and target table spaces or index spaces are not at the same version, you will need to run REPAIR VERSIONS on the target system after the MIGRATE is executed. If the indexes for a versioned space were not migrated, then these indexes would need to be rebuilt in a separate step on the target system.

Table 9 on page 283 describes the common dependencies for coding the MIGRATE command.
Table 9: MIGRATE command dependencies

<table>
<thead>
<tr>
<th>Option coded</th>
<th>Option that must be used</th>
</tr>
</thead>
<tbody>
<tr>
<td>FROMRBA or FROMLOGPOINT</td>
<td>LOGAPPLY ONLY</td>
</tr>
<tr>
<td>BACKOUT</td>
<td>TORBA or TOLOGPOINT</td>
</tr>
</tbody>
</table>

Table 10 on page 283 describes the common prohibitions to coding the MIGRATE commands.

Table 10: MIGRATE command prohibitions

<table>
<thead>
<tr>
<th>Option coded</th>
<th>Option that must not be used</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGONLY</td>
<td>TOCOPY</td>
</tr>
</tbody>
</table>

MIGRATE option descriptions

This section describes the options available with MIGRATE.

Not all of the options described are valid in a single MIGRATE command. The recovery strategy that best meets your needs determines which options you should include in a single command statement.

MIGRATE

Use the MIGRATE command when you want to perform a migration.

USING specification

The syntax diagram for the USING specification is in “MIGRATE syntax” on page 278.

USING dataSetName

The USING option is required when you specify the MIGRATE command. USING specifies the name of the migration file created by the COPY PLUS EXPORT command. The migration file contains migration information for one or more table spaces. This information includes object identifiers, table space and index space names, and other data relating to the target table spaces and index spaces.
CHANGE TABLE FROM creator1.tableName1 TO creator2.tableName2 or CHANGE INDEX FROM creator1.indexName1 TO creator2.indexName2

When the table names or index names are not the same between the source and the target subsystems, use the CHANGE ... FROM ... TO syntax to resolve differences.

Each table or index must contain a creator and a name. You can use an asterisk (*) as any portion of the name.

You must take care when using wildcarding. If the name in the TO clause does not produce a one-to-one match for all the tables or indexes on the target space, the space is considered unrecoverable. (The FROM name is the name in the migration file. For MIGRATE, that is the name on the target system. The TO name is the name on the source for MIGRATE.)

For example, you may want to copy the production payroll tables to the test subsystem, but the creator names between these two subsystems are different. Use the CHANGE TABLE feature to identify table name matches between the two subsystems. The migration file is searched for the FROM table name pattern and is substituted with the TO table name pattern. Multiple CHANGE ... FROM ... TO clauses are allowed in a single USING specification.

In the following specification, all the tables in the migration file with the creator name of PAYTEST resolve to a creator name of PAYPROD, which matches the tables on the source subsystem.

MIGRATE USING dataSetName CHANGE TABLE FROM PAYTEST.* TO PAYPROD.*

INDEXES

The INDEXES option specifies how RECOVER PLUS handles indexes when you specify the MIGRATE command. (You can also use INDEX or INDEXS in place of INDEXES.)

INDEXES RECOVER

If there are indexes in the migration data set, they will be recovered.

INDEXES NO

INDEXES NO specifies that RECOVER PLUS does not recover indexes that are present in the migration data set.

EXCLUDE (dbname.ts or qualifier.ix)

Use the EXCLUDE option in the USING specification to exclude one or more objects from migration. You can use wildcards or specific names to specify the exclusions. The following wildcards are valid:
- % and * match one or more characters in the object name.
- _ and ? match a single character in the object name.

List the excluded objects following the EXCLUDE keyword. Each item in the list must be in the form `dbname.ts` or `qualifier.ix` and you must separate the individual items with commas. Enclose the list in parentheses.

**DSNUM**

You can exclude multiple partitions from the specified object.

- DSNUM ALL is allowed with partitioned and nonpartitioned objects.
- DSNUM `begin: end` is allowed for partitioned objects to exclude a range of partitions.
- DSNUM `integer` is allowed for partitioned objects to exclude a single partition.

**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 are meaningless or misleading on the target DB2.

**TABLESPACE specification**

The syntax for the TABLESPACE specification is in Figure 42 on page 279.

**TABLESPACE `databaseName.tableSpaceName`**

The TABLESPACE option specifies the source table space to be migrated.

- `databaseName` is the name of the database containing the table space. The default is DSNDDB04. The name cannot be DSNDDB01, DSNDDB06, DSNDDB07, 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.
FROMRBA or FROMLOGPOINT specification

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 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 migrated. 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.

The syntax diagram for the FROMRBA or FROMLOGPOINT specification is in “MIGRATE syntax” on page 278.

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.

Note

If you use LASTQUIESCE and the quiesce is prior to the ALTER ADD PART, the migration 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 migration, 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.

**OBJECTSET specification**

The OBJECTSET option is available in MIGRATE statement syntax to migrate groups that are defined using RECOVERY MANAGER.

The OBJECTSET specification is shown in “MIGRATE syntax” on page 278.

**OBJECTSET creator.name**

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.

Following are restrictions that apply when you use OBJECTSET in any of its forms on the RECOVER statement:

- 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 clauses.

When you use OBJECTSET, if RECOVER PLUS tries to migrate 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 migration will complete.

If a table space is found in the OBJECTSET but it is not found in the migration file, the table space is considered unrecoverable. You can use the EXCLUDE specification to eliminate this object from the OBJECTSET.

**EXCLUDE (dbname.ts or qualifier.ix )**

Use the EXCLUDE option after an OBJECTSET specification to exclude one or more objects from the migration. You can use wildcards or specific names to specify the exclusions. The following wildcards are valid:

- % and * match one or more characters in the object name.
- _ and ? match a single character in the object name.

List the excluded objects following the EXCLUDE keyword. Each item in the list must be in the form *dbname.ts* or *qualifier.ix* and you must separate the individual items with commas. Enclose the list in parentheses.

**INDEPENDENT OUTSPACE specification**

The syntax diagram for the INDEPENDENT OUTSPACE specification is in “MIGRATE syntax” on page 278.
INDEPENDENT OUTSPACE or INDEP OUTSPACE

Use the INDEPENDENT (or INDEP) OUTSPACE option to redirect the output of the migration to a data set that is not used by the source DB2. INDEP OUTSPACE migrations do not require you to stop the source spaces that are being migrated, and they do not interfere with production processing. They do require that you stop the target DB2 spaces during migration. INDEP OUTSPACE enables you to do the following things:

If you use TABLESPACE keyword, you can specify the name of each independent data set or you can use a model name to represent the VSAM data set names of several target table spaces.

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.
- If the target 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 85 and “Secondary extents” on page 85.

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, nonpartitioned spaces. For example, RECOVER PLUS will dynamically allocate A002, A003, to A00n INDEP OUTSPACE data sets as needed. For partition-by-growth table spaces, you must ALTER add partitions to the target space for data sets that do not exist.

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 “STOGROUP specification description” on page 123). RECOVER PLUS statements that use STOGROUP... USEORDER are shown in the following example:

```plaintext
OPTIONS STOGROUP TESTSTG USEORDER(NEW004,NEW005,NEW006)
```

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:
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 “MIGRATE syntax” on page 278.

**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 the symbolic variables in “Using symbolic variables” on page 510.

**TORBA or TOLOGPOINT specification**

The syntax diagram for the TORBA or TOLOGPOINT specification is in “MIGRATE syntax” on page 278.
**TORBA or TOLOGPOINT**

Use TORBA or TOLOGPOINT to migrate the image of 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 MIGRATE, see “MIGRATE command dependencies and prohibitions” on page 282.)

You can use TORBA or TOLOGPOINT interchangeably, regardless of the version of DB2 that you are using.

A TOLOGPOINT migration creates an image of 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.

*Note*

Use of MIGRATE with TORBA or TOLOGPOINT is not a point-in-time recovery. It is simply creation of an image as of a prior point in time.

**TORBA LASTQUIESCE (relativeGenerationNumber) or TOLOGPOINT LASTQUIESCE (relativeGenerationNumber)**

Use TORBA LASTQUIESCE or TOLOGPOINT LASTQUIESCE to migrate to the most recent quiesce point registered in SYSIBM.SYSCOPY for the table space being migrated.

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 quiesce. For example, TOLOGPOINT LASTQUIESCE (-1) indicates the use of the quiesce before the most recent quiesce.

If you are migrating multiple table spaces in a single MIGRATE command statement, all of the table spaces must have a common last quiesce point. However, if you are using multiple MIGRATE command statements to migrate multiple table spaces, each table space can have a different last quiesce point.

LASTQUIESCE can be abbreviated as LASTQ.

*Note*

If you use LASTQUIESCE and the quiesce is prior to an ALTER ADD PART, the migration 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 migrate a set of table spaces and indexes to the most recent common quiesce point.
point registered in SYSIBM.SYSCOPY. 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 is migrated, the effect is exactly the same as using TORBA LASTQUIESCE.

TORBA LASTARCHQ or TOLOGPOINT LASTARCHQ

Use TORBA LASTARCHQ or TOLOGPOINT LASTARCHQ to migrate to the log point of the last ARCHIVE LOG MODE(QUIESCE) command issued for the DB2 subsystem.

TORBA LASTSHUTDOWN or TOLOGPOINT LASTSHUTDOWN

Use TORBA LASTSHUTDOWN or TOLOGPOINT LASTSHUTDOWN to migrate 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 migrate 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 migrate 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 to be migrated 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 migrate 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.

**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.

**TORBA X'logPoint' or TOLOGPOINT X'logPoint'**

Use TORBA X'logPoint' or TOLOGPOINT X'logPoint' to migrate to a prior point in time identified by the log point, 'logPoint'. 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.

**TOCOPY LASTCOPY specification**

The syntax diagram for the TOCOPY specification is in "MIGRATE syntax" on page 278.

**TOCOPY**

The TOCOPY option specifies the last image copy to be processed by RECOVER PLUS to migrate the specified table space (full or incremental copies). The migration process ends after this image copy has been applied. If you specify an incremental copy for table space migration, the prior full copy and any intervening incremental copies are also used in the migration process. The image copy specified must be registered in SYSIBM.SYSCOPY 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 LASTCOPY, the log is not used to apply updates that are made after the copy.

After a TOCOPY LASTCOPY migration, 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 LASTCOPY migration is no longer reflected in the space but may be reinstated by a subsequent TOLOGPOINT recovery.
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.)

A point-in-time recovery event is not recorded in SYSIBM.SYSCOPY or BMCXCOPY.

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 is the last one to be processed by RECOVER PLUS to migrate 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.

LOGSORT specification

For a description of the LOGSORT specification options, see “LOGSORT work data set specification description” on page 119. The syntax diagram for the LOGSORT specification is in “MIGRATE syntax” on page 278.

LOGAPPLY ONLY

LOGAPPLY ONLY specifies that only log information is to be used to migrate a table space 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 restored one or more target table spaces 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 in a single MIGRATE command statement, you need to specify only a single LOGAPPLY ONLY. However, you must specify FROMRBA or FROMLOGPOINT for each table space.

When you use LOGAPPLY ONLY, you cannot use the TOCOPY LASTCOPY 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 on multiple table spaces.

When you use LOGONLY, you cannot use the TOCOPY LASTCOPY option.

CLONE

The CLONE option indicates that RECOVER PLUS is to migrate clone table data. When you migrate 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 RECOVER PLUS 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 RECOVER PLUS step.

ANALYZE

The ANALYZE option prints a RECOVER PLUS 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 PLUS 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 specified 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 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 run. You cannot restart a RECOVER PLUS job that specifies ANALYZE ONLY, but you can start a new job.

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

If you provide a SYSPICK DD statement, RECOVER PLUS generates a list of all of the input tape and cartridge volumes that are allocated. For more information about SYSPICK, see “RECOVER PLUS data sets and RECOVER PLUS DD statements” on page 325.

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

**TOLOCATION**

The TOLOCATION option improves the handling of partition-by-growth spaces with more than one data set by allowing RECOVER PLUS to automatically alter the number of data sets on the target system.
location is the location name that can be used in the CONNECT SQL statement to connect to a target subsystem.

**IMPORT**

The IMPORT command allows you to move data, by use of image copies, from one or more table spaces to another.

A migration file that the COPY PLUS EXPORT command creates facilitates the migration. You run the EXPORT command on the source subsystem to collect information on one or more table spaces and index spaces. This information includes object identifiers, the names of image copies, and other information relating to the source table spaces and index spaces. You run the RECOVER PLUS IMPORT command on the target subsystem to migrate all the image copy data to the target table spaces and index spaces that are identified by the command.

The IMPORT command uses the object identifiers found in the migration file and translates them to the object identifiers found in the target catalog.

*Note*

This command requires one of the following valid passwords:

- a Recovery Management solution password
- a Database Administration solution password

The Database Administration (DAD) password allows RECOVER PLUS to run within a worklist, but does not include full-function RECOVER PLUS.

For more information about copy migration using COPY PLUS EXPORT and RECOVER PLUS IMPORT, see the *Recovery Management User Guide*.

**IMPORT syntax**

The following figures show the syntax of the IMPORT command.
Figure 49: IMPORT syntax

IMPORT \[^{1,2}\]  USING specification

- TABLESPACE
  - DSNDB04
  - databaseName.
- tableSpace Name
- AUX
  - NO
  - ALL
  - XML
  - LOB
- HISTORY
- RICHK
  - NO
  - YES

OBJECTSET specification

INDEPENDENT OUTSPACE specification

OUTCOPY specification

- CLONE
- ANALYZE
  - YES
  - NO
  - ONLY

REDEFINE
- YES
- NOSCATCH

REUSE

UPDATE VERSIONS

STATS
- NO
- YES

DEFINE
- NO
- YES

\[^{1}\] Requires a valid password as follows: Recovery Management or Database Administration
\[^{2}\] Not valid with BACKOUT
Figure 50: IMPORT syntax—USING specification

**USING specification**

```
IMPORT

USING dataSetName

CHANGE TABLE FROM creator1.tableName1 TO creator2.tableName2

CHANGE INDEX FROM creator1.indexName1 TO creator2.indexName2

EXCLUDE ()

EXCLUDE (dbName.ts qualifier.ix)

EXCLUDE (DSNUM ALL integer begin: end)

SYNC AUTO REPLACE INDEXES AUTO RECOVER REBUILD NO

RESET NOCOPYPEND
```

Figure 51: IMPORT syntax—OBJECTSET specification

**OBJECTSET specification**

```
IMPORT

OBJECTSET creator.name

EXCLUDE ()

EXCLUDE (dbName.ts qualifier.ix)
```

Chapter 3  RECOVER PLUS syntax  299
IMPORT command dependencies and prohibitions

Following are command dependencies and prohibitions for IMPORT:

- INCOPY is assumed and therefore is not allowed on the IMPORT command. OBID translation is also assumed.

- Since the IMPORT command requires the use of image copies, BACKOUT cannot be specified.

- When coding OUTCOPY and INDEP OUTSPACE, the output image copies cannot be registered.

- When coding INDEP OUTSPACE, REDEFINE is ignored.

The IMPORT command has the following limitations:

- no support for incremental copies
- no support for copies of the DB2 catalog
- no support for specifying DSNUM

When selecting image copies, the IMPORT command uses a DSNUM 0 copy or DSNUM n copies. DSNUM n copies will only be considered when all the partitions or data sets are registered at the same log point.

**IMPORT option descriptions**

This section describes the options available with IMPORT.

Not all of the options described are valid in a single IMPORT command. The recovery strategy that best meets your needs determines which options you should include in a single command statement.

**IMPORT**

Use the IMPORT command when you want to perform a migration.

**USING specification**

The syntax diagram for the USING specification is in “IMPORT syntax” on page 297.

**USING dataSetName**

The USING option is required when you specify IMPORT command. USING specifies the name of the migration file created by the COPY PLUS EXPORT command. The migration file contains migration information for one or more table spaces and index spaces. This information includes object identifiers, copy data set names, and other data relating to the target table spaces and index spaces.

**CHANGE TABLE FROM creator1.tableName1 TO creator2.tableName2 or CHANGE INDEX FROM creator1.indexName1 TO creator2.indexName2**

When the table names or index names are not the same between the source and the target subsystems, use the CHANGE ... FROM ... TO syntax to resolve differences.

Each table or index must contain a creator and a name. You can use an asterisk (*) as any portion of the name.
You must take care when using wildcarding. If the name in the FROM clause does not produce a one-to-one match for all the tables or indexes on the target space, the space is considered unrecoverable.

For example, you may want to copy the production payroll tables to the test subsystem, but the creator names between these two subsystems are different. Use the CHANGE TABLE feature to identify table name matches between the two subsystems. The migration file is searched for the FROM table name pattern and is substituted with the TO table name pattern. Multiple CHANGE TABLE FROM ... TO clauses are allowed in a single USING specification.

In the following specification, all the tables in the migration file with the creator name of PAYPROD resolve to a creator name of PAYTEST, which matches the tables on the target subsystem.

```sql
IMPORT USING dataSetName
CHANGE TABLE FROM PAYPROD.* TO PAYTEST.*

EXCLUDE (dbname.ts or qualifier.ix)
```

Use the EXCLUDE option in the USING specification to exclude one or more objects from the import process. You can use wildcards or specific names to specify the exclusions. The following wildcards are valid:

- `%` and `*` match one or more characters in the object name.
- `_` and `?` match a single character in the object name.

List the excluded objects following the EXCLUDE keyword. Each item in the list must be in the form `dbname.ts` or `qualifier.ix` and you must separate the individual items with commas. Enclose the list in parentheses.

```sql
EXCLUDE (dbname.ts or qualifier.ix)
```

**DSNUM**

You can exclude multiple partitions from the specified object.

- `DSNUM ALL` is allowed with partitioned and nonpartitioned objects.
- `DSNUM begin: end` is allowed for partitioned objects to exclude a range of partitions.
- `DSNUM integer` is allowed for partitioned objects to exclude a single partition.

**SYNC**

Use SYNC to indicate if the migration should include all spaces in the IMPORT command or only those spaces that have changed since the last migration.
SYNC is useful in cases where the image copy indicated by the current EXPORT file is identical to the spaces imported previously.

**SYNC AUTO**

Specify SYNC AUTO (the default) to replace only spaces that have changed since the last time those spaces were imported. Consider this option when migrating to a query-based environment when the spaces are read only. Cost savings can be significant if you only refresh the objects that have changed since the last migration.

**SYNC REPLACE**

Specify SYNC REPLACE to replace all spaces regardless of whether they have changed or not. Consider this option when migrating to a subsystem where the data is manipulated in test environment.

**INDEXES**

The INDEXES option specifies how RECOVER PLUS handles indexes when you specify the IMPORT command. (You can also use INDEX or INDEXS in place of INDEXES.)

**INDEXES AUTO**

INDEXES AUTO, the default value, specifies that RECOVER PLUS recovers indexes if information is available for them in the migration data set. Any index not in the migration data set but defined on the target system is rebuilt.

**INDEXES RECOVER**

INDEXES RECOVER specifies that indexes on the target system are recovered if they have entries in the migration data set.

**INDEXES REBUILD**

INDEXES REBUILD specifies that all indexes on the target system are rebuilt regardless of whether they have entries in the migration data set.

**INDEXES NO**

INDEXES NO specifies that indexes on the target system are left unchanged.

**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 are meaningless or misleading on the target DB2.
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.

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.

Because the input copy is not registered on the target system, the target space is not recoverable after the IMPORT. You cannot run a RECOVER to current on the target system because no usable copies are registered in SYSCOPY or BMCXCOPY (but you could run IMPORT again).

**TABLESPACE** specification

The TABLESPACE option specifies the target table space to be migrated.

The syntax diagram for the TABLESPACE specification is in Figure 49 on page 298.

**TABLESPACE** `databaseName.tableSpaceName`

If you do not specify the TABLESPACE option, all of the table spaces in the migration file will be imported. If you specify the TABLESPACE option, only the specified table spaces will be imported.

- `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.

**AUX**

The AUX option specifies if auxiliary objects will be included with the migration of the base table spaces.

**Note**

If you specify the AUX keyword and the related table spaces are not included in the migration file, the migration will fail.
If you do not specify AUX on one of these commands, RECOVER PLUS uses the value of the AUX installation option (“AUX=NO” on page 608), which defaults to NO.

**AUX NO**

RECOVER PLUS does not include any auxiliary objects in the migration.

**AUX ALL**

RECOVER PLUS includes XML and LOB objects in the migration.

**AUX XML**

RECOVER PLUS includes all XML data objects along with the XML base table space. All implicitly created XML table spaces will be processed along with the base XML table space.

**AUX LOB**

RECOVER PLUS includes all LOB data objects along with the LOB base table space. All table spaces created with the LOB attribute will be processed along with the base LOB table space.

**HISTORY**

The HISTORY option specifies if history tables will be included in the migration. The AUX keyword must be specified in order to use the HISTORY keyword.

The meaning of the HISTORY keyword depends on which AUX option you specify, as follows:

- **AUX NO HISTORY** includes regular tables, system-time base tables, and their related HISTORY tables. It does not include related XML or LOB objects.

- **AUX ALL HISTORY** includes regular tables, system-time base tables and their related HISTORY tables, and all related XML or LOB objects.

- **AUX XML HISTORY** includes regular tables, system-time base tables and their related HISTORY tables, and all related XML objects.

- **AUX LOB HISTORY** includes regular tables, system-time base tables and their related HISTORY tables, and all related LOB objects.

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 migrated as a set.

- The system-maintained temporal space and the history space migrate to the same log point.
- All DSNUMs are migrated in the same SYSIN.

**RICHK**

RICHK specifies if RECOVER PLUS will check for referential integrity constraints on the objects being recovered.

**RICHK YES**

RICHK YES checks for referential integrity constraints and sets check pending (CHKP) status for dependent objects not recovered in the same step to the same PIT.

**RICHK NO**

RICHK NO does not check for referential integrity. NO is the default value.

**OBJECTSET specification**

The OBJECTSET option is available in IMPORT statement syntax to import groups that are defined using RECOVERY MANAGER. If you do not specify the TABLESPACE or OBJECTSET options, all of the table spaces in the migration file will be imported. If you specify the OBJECTSET option, only the specified table spaces will be imported.

The OBJECTSET specification is shown in “IMPORT syntax” on page 297.

**OBJECTSET creator.name**

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 in any of its forms on the RECOVER statement:

- 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 clauses.

When you use OBJECTSET, if RECOVER PLUS tries to migrate 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 migration will complete.

If a table space is found in the OBJECTSET but it is not found in the migration file, the table space is considered unrecoverable. You can use the EXCLUDE specification to eliminate this object from the OBJECTSET.

EXCLUDE (dbname.ts or qualifier.ix)

Use the EXCLUDE option after an OBJECTSET specification to exclude one or more objects from migration. You can use wildcards or specific names to specify the exclusions. The following wildcards are valid:

- % and * match one or more characters in the object name.
- _ and ? match a single character in the object name.

List the excluded objects following the EXCLUDE keyword. Each item in the list must be in the form dbname.ts or qualifier.ix and you must separate the individual items with commas. Enclose the list in parentheses.

INDEPENDENT OUTSPACE specification
The syntax diagram for the INDEPENDENT OUTSPACE specification is in "IMPORT syntax" on page 297.

**INDEPENDENT OUTSPACE or INDEP OUTSPACE**

Use the INDEPENDENT (or INDEP) OUTSPACE option to redirect the output of the migration to a data set that is not used by DB2. INDEP OUTSPACE does not interfere with production processing. INDEP OUTSPACE enables you to do the following things:

You can specify the name of each independent data set or you can use a model name to represent the VSAM data set names of several target table spaces.

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.

- If the target 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 85 and “Secondary extents” on page 85.

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 “STOGROUP specification description” on page 123). RECOVER PLUS statements that use STOGROUP... USEORDER are shown in the following example:

```
OPTIONS STOGROUP TESTSTO USEORDER(NEW004,NEW005,NEW006)
```

The default model name is identical to the DB2 name of the object being recovered, but with BMCDBC substituted for DSNDBC 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.

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 “Using symbolic variables” on page 510.

OUTCOPY specification

For option descriptions for the OUTCOPYDDN and RECOVERYDDN specifications, which are included in the OUTCOPY specification, see

- “OUTCOPYDDN specification” on page 220
- “RECOVERYDDN specification” on page 221
The syntax diagram for the OUTCOPY specification is in “IMPORT syntax” on page 297.

**OUTCOPY**

You can specify OUTCOPY to make copies of the object or object data set being imported. 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 migrating 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. However, if you are using the INDEPENDENT OUTSPACE option to import to non-DB2 data sets and you request image copies, you cannot register those copies.

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

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 220 and “RECOVERYDDN specification” on page 221. The copies that you want to register are specified by the REGISTER option (“OUTCOPY specification” on page 212).

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 (“OUTCOPY specification” on page 212).

---

**Note**

You can use RECOVER PLUS to automatically size the OUTCOPY files generated by RECOVER PLUS. For more information, see “Automatic sizing of output copies dynamically allocated on DASD” on page 514.

---

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 127.

- If the object is nonpartitioned and you are importing multiple data sets by specifying DSNUM ALL, each output copy is for all of the data sets. If you
are importing 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 being imported. 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.

OUTCOPYDDN specification

The syntax diagram for the OUTCOPYDDN specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

RECOVERYDDN specification

The syntax diagram for the RECOVERYDDN specification is in “RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax” on page 161.

CLONE

The CLONE option indicates that RECOVER PLUS is to import clone table data. When you import 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 RECOVER PLUS 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 RECOVER PLUS step.

ANALYZE

The ANALYZE option prints a RECOVER PLUS 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 PLUS 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 specified 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 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 run. You cannot restart a RECOVER PLUS job that specifies ANALYZE ONLY, but you can start a new job.
Use the ANALYZE ONLY option to determine which phases will occur with a specific request, or which copy data sets and log data sets will be used.

If you provide a SYSPICK DD statement, RECOVER PLUS generates a list of all of the input tape and cartridge volumes that are allocated. For more information about SYSPICK, see “RECOVER PLUS data sets and RECOVER PLUS DD statements” on page 325.

If you specify OPTIONS EARLYRECALL with ANALYZE ONLY, the recall of the data sets needed for execution 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 re allocation 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 PLUS command. To process multiple STOGROUP-defined objects with some reallocated and others not reallocated, you must use one RECOVER PLUS command statement for the objects to be reallocated and a second RECOVER PLUS 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 execution. For more information, see “Storage group-defined data sets” on page 84.

NOSCRATCH

If you specify NOSCRATCH with REDEFINE YES, RECOVER PLUS invokes IDCAMS to uncatalog without scratching.

Using this option eliminates mount messages for the primary site packs that do not exist at the recovery site.
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 190. 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 or index 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.

If you use RECOVER PLUS and this option to perform migration, you need to

- make sure that your copies contain system pages or directory pages for all versions of the space

- import table spaces in a separate step from rebuilding indexes
  You can recover indexes and tables spaces in the same step, but rebuilds must be in a separate step.
STATS

The STATS option allows you to carry forward existing RUNSTATS from a migration file. These RUNSTATS may then be optionally applied during a RECOVER PLUS IMPORT. To import stats, STATS YES must have been specified in the COPY PLUS EXPORT command.

SYSINDEXSTATS, SYSLOBSTATS, and SYSTABSTATS are included in the migration file. Real-time statistics are included in a migration, regardless of the STATS specification.

STATS YES

STATS YES updates RUNSTATS information on the target system using information from the migration file. RTS stats are also updated.

STATS NO

RUNSTATS information is not updated. RTS stats are always updated. STATS NO is the default.

DEFINE

DEFINE specifies if DEFINE NO objects will be instantiated.

Note

DB2 Version 11 or later supports the DEFINE option.

DEFINE YES

RECOVER PLUS instantiates DEFINE NO objects if necessary, and updates the catalog and DBD to reflect that the object has been defined.

DEFINE NO

RECOVER PLUS does not process DEFINE NO objects. NO is the default value.
Building and executing RECOVER PLUS jobs

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 ("JOB statement" on page 318)
- an EXEC statement with the appropriate utility parameters ("EXEC statement" on page 318 and "Utility parameters on the EXEC statement" on page 319)
- STEPLIB or JOBLIB DD statements as needed ("STEPLIB DD or JOBLIB statements" on page 325)
- DD statements as needed for the data sets needed for the recovery ("RECOVER PLUS data sets and RECOVER PLUS DD statements" on page 325 to "Determining unloaded keys file space requirements" on page 341)
- RECOVER PLUS statements using the appropriate command syntax ("Specifying RECOVER PLUS syntax" on page 341)
A simple example of a RECOVER PLUS job is illustrated in the following figure. For more examples of RECOVER PLUS jobs, see “Examples of RECOVER PLUS jobs” on page 377.

**Figure 54: JCL for a simple RECOVER PLUS execution**

```
//DMBRECV2 JOB (5212), 'SIMPLE2', CLASS=A, MSGCLASS=X, NOTIFY=AFR
//BMCRCVR EXEC PGM=AFRMAIN, REGION=0M, PARM="DB21, RDADBEX2, 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” on page 318 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 319.

You can include the REGION parameter on either your EXEC statement or your JOB statement. See “REGION parameter” on page 318 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 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 2 GB. For more information, see “Setting the MEMLIMIT parameter” on page 70.

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 474.

### 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:

```bash
//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:

```bash
// 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 342.

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 “Common utility tables” on page 621.) 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 342.

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 342.

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 637.

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 11 on page 326. For examples of the different output files and their content, see “Using MSGLEVEL to control RECOVER PLUS output” on page 348.

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 83 and “RECOVER PLUS installation options” on page 587.

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 “RECOVER PLUS installation options” on page 587.

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 “RECOVER PLUS installation options” on page 587.
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 78.

**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.

*Note*

If RECOVER UNLOADKEYS was specified for an index, the associated table space is placed in RO status.

**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.
As an alternative to `afrOpts`, 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 “RECOVER PLUS installation options” on page 587.

**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 installation options module) 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 74.)

**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 348.

The following table 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 11: RECOVER PLUS data sets

<table>
<thead>
<tr>
<th>Data set name</th>
<th>Required or optional</th>
<th>Dynamically allocated (yes or no)</th>
<th>MSGLEVEL value required</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. 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 “RECOVER PLUS syntax” on page 87. Unicode is not supported in the SYSIN file.</td>
</tr>
<tr>
<td>Data set name</td>
<td>Required or optional</td>
<td>Dynamically allocated (yes or no)</td>
<td>MSGLEVEL value required</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------</td>
<td>-----------------------------------</td>
<td>-------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>AFRPRINT</td>
<td>required</td>
<td>yes</td>
<td>1 or 2</td>
<td>contains a list of the phases executed for each object and the name of the AFRPR nnn data set where RECOVER PLUS put the output for the phase</td>
</tr>
<tr>
<td>AFRPR nnn</td>
<td>required</td>
<td>yes</td>
<td>0, 1, or 2</td>
<td>provides messages for each execution phase RECOVER PLUS allows up to (MAXKSORT + MAXLSORT + 1) AFRPR nnn files. RECOVER PLUS names the files AFRPR001, AFRPR002, AFRPR003, and so on.</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 e</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>AFRPLAN e</td>
<td>optional, based on MSGLEVEL</td>
<td>yes</td>
<td>2</td>
<td>records the execution plan When you use a Recovery Management for DB2 solution password, AFRPLAN includes values for recovery estimation.</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>Data set name</td>
<td>Required or optional</td>
<td>Dynamically allocated (yes or no)</td>
<td>MSGLEVEL value required</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------</td>
<td>-----------------------------------</td>
<td>-------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| AFRTIME          | optional             | yes, if needed                    | not applicable          | 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                                                                                                                                                                                                                                                                                                                                 |
| AFRTRACE         | not applicable       | yes                               | not applicable          | provides trace information when errors occur during execution                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| SYSERR or AFRDUMP| not applicable       | yes                               | not applicable          | 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.                                                                                                                                                                                                                                                                                                                                             |
| SYSOU nnn        | required             | yes                               | not applicable          | used for BMCSORT message output                                                                                                                                                                                                                                                                                                                                                                                                              |
| SYSUDUMP         | not applicable       | yes                               | not applicable          | 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.                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| SYSSCAN          | optional             | no                                | not applicable          | contains the output from the LOGSCAN command
If the SYSSCAN DD is not specified, the LOGSCAN output is in AFRPRINT.                                                                                                                                                                                                                                                                                                                                                                                                                                   |
<table>
<thead>
<tr>
<th>Data set name</th>
<th>Required or optional</th>
<th>Dynamically allocated (yes or no)</th>
<th>MSGLEVEL value required</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 509.</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. Allocation of the SYSPICK data set DCB should be RECFM=VB,LRECL=137.</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>
<tr>
<td>Data set name</td>
<td>Required or optional</td>
<td>Dynamically allocated (yes or no)</td>
<td>MSGLEVEL value required</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------</td>
<td>-----------------------------------</td>
<td>-------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>a</strong></td>
<td></td>
<td></td>
<td></td>
<td>The values shown are the required values of MSGLEVEL to produce the output file. The default value of MSGLEVEL is 1.</td>
</tr>
<tr>
<td><strong>b</strong></td>
<td></td>
<td></td>
<td></td>
<td>When &quot;not applicable&quot; 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.</td>
</tr>
<tr>
<td><strong>c</strong></td>
<td></td>
<td></td>
<td></td>
<td>For descriptions of the RECOVER PLUS commands and options, see Chapter 3, &quot;RECOVER PLUS syntax.&quot;. Sections are included with information on the use of Long Names and Unicode.</td>
</tr>
<tr>
<td><strong>d</strong></td>
<td></td>
<td></td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td><strong>e</strong></td>
<td></td>
<td></td>
<td></td>
<td>RECOVER PLUS produces these data sets regardless of the value of MSGLEVEL when you specify OPTIONS ANALYZE ONLY.</td>
</tr>
</tbody>
</table>

**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 325.

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 “Examples of RECOVER PLUS jobs” on page 377.

**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.

**Note**

Tape stacks with more than 255 volumes are not supported. You must split the recovery into two or more steps when using such a tape stack.

### 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 507).

**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:

  $DDName_{nn}$ or $DDName_{nnn}$

  $DDName$ 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.

  The following table lists the default ddname prefixes.
Table 12: 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>BMCRSZ</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 the preceding table 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 the preceding table 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 the preceding table 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 the preceding table 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 126).
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 336.

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 326.

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 594 and “INDEXLOG” on page 134.) 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 335.

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 SORTNUM
installation option or the OPTIONS SORTNUM parameter.

SYSOUunn: sort message data sets
unn 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 143, “MAXKSORT” on page 597, “Setting the MAXKSORT option” on
page 495, and “Understanding the UNLOADKEYS and BUILDINDEX
strategy” on page 478.

**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 336.

**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 “RECOVER PLUS data sets and RECOVER PLUS DD statements” on page 325). 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 415.

**Work data set needed for extracted keys (SKEYDDN)**

This data set holds keys for subsequent RECOVER BUILDINDEX 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 BUILDINDEX 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 336.

DD statements for RECOVER BUILDINDEX jobs

In addition to those common to all RECOVER PLUS executions, DD statements must be included in the JCL when you specify a RECOVER BUILDINDEX command statement. Use DD statements for work data sets for extracted keys (SKEYDDN).

For examples of the DD statements used in a RECOVER BUILDINDEX job, see “Example 7: Building a nonpartitioned index” on page 415. For DD statements common to all RECOVER PLUS executions, see “DD statements common to all RECOVER PLUS executions” on page 326.

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.

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.

— LOGOUnnn 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 “SMCORE=(0K,0K)” on page 599)

- 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 “RECOVER PLUS installation options” on page 587

- number of parallel index key sorts, which is controlled by the MAXKSORT option (“MAXKSORT integer ” on page 143 and “MAXKSORT” on page 597) and the KSORTSHARE option (“MAXKSORT integer ” on page 143 and “KSORTSHARE=YES” on page 607)

- 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 489 and “Understanding the UNLOADKEYS and BUILDINDEX strategy” on page 478.
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:

```
S001WK01 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:

```
BMC96308I ESTIMATED NUMBER OF TRACKS FOR ALL INDEX KEY SORT WORK DATA SETS
FOLLOWS:
  BMC96309I  S001WK**:       5627  (TRACKS)
  BMC96309I  S002WK**:       5627  (TRACKS)
  BMC96309I  S003WK**:       9184  (TRACKS)
  BMC96309I  S004WK**:       9184  (TRACKS)
  BMC96309I  S005WK**:       9184  (TRACKS)
  BMC96309I  S006WK**:       5627  (TRACKS)
```

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 10 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 10 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 S020WK**.

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 ("MAXLSORT integer " on page 145 and "MAXLSORT=0" on page 606) 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:

\[ \text{Size} = NR \times RL \]

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.
- **RL** is the record length displayed in message BMC40806 LENGTH OF RECORDS=length. This message prints in the RECOVER PLUS plan when SKEYDDN is specified and can be obtained from an ANALYZE ONLY job. This length is the maximum key size involved plus six or seven bytes. (For more details, see “Index sort work space requirements” on page 338.)

For unloaded keys files, all partitions in a run are added together.

Specifying RECOVER PLUS syntax

Place recovery commands in the SYSIN data set.

Use “RECOVER PLUS syntax” on page 87 and “Examples of RECOVER PLUS jobs” on page 377 as guides to the syntax that is required for completion of your job setup.

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 71.

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 348. A RECOVER UNLOADKEYS job also requires some cleanup as discussed in “Cleaning up after a RECOVER UNLOADKEYS job” on page 348.

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 325.

If you are not using SIMULATE YES option or the RECOVER PLUS redirection option, INDEP OUTSPACE, the table spaces or indexes that are specified in your RECOVER PLUS commands must have an acceptable status before the job will run. For information about unacceptable table and index space status, see “Initial table space and index space status” on page 75.

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 (“Restart parameter (restartParm)” on page 320) 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.

The following table shows the kinds of restart that are possible for each phase of RECOVER PLUS.

Table 13: 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</td>
<td></td>
</tr>
<tr>
<td>RESTORE</td>
<td>yes</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>yes</td>
</tr>
<tr>
<td>BUILD</td>
<td>b</td>
</tr>
<tr>
<td>UTILTERM</td>
<td></td>
</tr>
</tbody>
</table>

1 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.

2 A phase restart from the BUILD phase is possible only when you specify WORKDDN with REBUILD INDEX. If you specify NOWORKDDN, the restarted run performs an UNLOAD phase, and then a BUILD phase.

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 \texttt{DISP=(MOD, CATLG, CATLG)} or \texttt{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 \texttt{(+ n)} value to \texttt{(+ n-m)}, where \texttt{m} is the relative generation number for the last cataloged generation in the original execution.

- For cataloged stacked tape copies, remove \texttt{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.

- Failure to remove \texttt{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 \texttt{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. \textit{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 \texttt{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 507.)
**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 “RECOVER PLUS installation options” on page 587.

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 322.

**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 in “CHECKINT (integer)” on page 122.
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 626.

**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 “BMCUTIL table” on page 642.

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 637.

### 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.

Valid values of MSGLEVEL are 0, 1, and 2, with a default value of 1 (see “Message level parameter (msgLevel)” on page 322). The example in the following figure 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 11 on page 326.

**Figure 55: RECOVER PLUS JCL for recovery of table space and index**

```plaintext
//AFREXMLx JOB (PAFR), 'EXAMPLE MSGVLx', // CLASS=Q, NOTIFY=&SYSUID, // MSGCLASS=X
```
Sample output with MSGLEVEL(0)

When you specify MSGLEVEL(0), RECOVER PLUS produces the following output files:

- AFRSUMRY (Figure 56 on page 349)
- AFRSTMT (Figure 57 on page 350)
- AFRPR001 (Figure 58 on page 352)

Figure 56: AFRSUMRY with MSGLEVEL(0)

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884 7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40001I UTILITY EXECUTION STARTING 08/28/2014 10:45:51
BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40876I MAINT: NO RECOVER PLUS PTFS APPLIED
BMC96010I SOLUTION COMMON CODE V11.2.01
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFS APPLIED
BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BJ0195 BJ0198 BJ0209 BJ0250 BJ0263 BJ0269 BJ0279
BMC40876I MAINT: BJ0288 BJ0308 BJ0343 BJ0357 BJ0361 BJ0374 BJ0390
BMC40876I MAINT: BJ0404 BJ0473 BJ0503 BJ0521

BMC40001I UTILITY ID = EXAMPLE11, DB2 SUBSYSTEM ID = DEFQ,
BMC40094I ATTEMPTING TO CONNECT TO DB2 SUBSYSTEM DEFQ, USING PLAN AFRB111T
BMC40024I SUCCESSFUL CONNECT TO DEFQ (RELEASE 1010 OF DB2) USING PLAN AFRB111T
BMC40475I INPUT STATEMENTS PRINTED IN AFRSTMT
BMC40336I ANALYZE FINISHED.
BMC40287I ELAPSED TIME = 00:00:03, UTILITY ELAPSED TIME = 00:00:04
Using MSGLEVEL to control RECOVER PLUS output

Figure 57: AFRSTMT with MSGLEVEL(0)

```plaintext
BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884 7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40931I PARM LIST:  DB2 SSID = DEFQ
BMC40932I              UTILID = EXAMPLE11
BMC40933I               RESTART = NEW/RESTART
BMC40934I                MSGLEVEL = 0
BMC40930I                PARMLIST NOT CODED - CHECKPT
BMC40931I                PARMLIST NOT CODED - RDB2STAT
BMC40930I                PARMLIST NOT CODED - AFRROPTS
BMC40930I                PARMLIST NOT CODED - ACAOPTS
BMC40931I                OPTIONS:
BMC40942I              PLANRECV = AFRB1111T       (MAIN PLAN)
BMC40941I              BINDQUALIFIER = BMCAFR       (QUALIFIER USED FOR DYNAMIC BIND)
BMC40241I              PUBLICPLAN = YES            (GRANT PLAN TO PUBLIC)
BMC96003I              ZIIP = ENABLED        (ZIIP PROCESSOR ENABLEMENT)
BMC40957I              INDEXLOG = NO             (DEFAULT USE OF INDEXLOG STRATEGY)
BMC40981I              IXRECP = NO             (SET INDEX RECOVER PENDING)
BMC96245I              KSORTSHARE = YES            (KEY SORTS SHARED BY ALL TASKS)
BMC40942I              MAXKSORT = 10  (DEFAULT)  (MAX CONCURRENT INDEX KEY SORTS)

BMC40933I                  ZIIP = ENABLED        (ZIIP PROCESSOR ENABLEMENT)
BMC40941I              BACKOUT = AUTO            (DEFAULT POINT-IN-TIME RECOVERY)

BMC40930I                OPTIONS:
BMC40957I              INDEXLOG = NO             (DEFAULT USE OF INDEXLOG STRATEGY)
BMC40981I              IXRECP = NO             (SET INDEX RECOVER PENDING)
BMC96245I              KSORTSHARE = YES            (KEY SORTS SHARED BY ALL TASKS)
BMC40942I              MAXKSORT = 10  (DEFAULT)  (MAX CONCURRENT INDEX KEY SORTS)

BMC40931I                  OPTIONS:
BMC40941I              OPNDB2ID = YES            (ACQUIRE RACF AUTHORITY OF DB2)
BMC40941I              RDB2STAT = YES            (RESET DB2 OBJECT STATUS)
BMC40941I              AMSCAT = NO             (USE 'CATALOG' IN IDCAMS INPUT)
BMC40931I              WKUNIT = SYSALLDA       (WORK UNIT NAME)
BMC40987I              WTOR = YES            (ISSUE WTOR IF SPACE REMAINS IN STOPP

BMC40933I                  OPTIONS:
BMC40941I              USEHDROBIDS = YES          (USE HEADER PAGE IDS FOR OBIDXLAT)
BMC40931I                  OPTIONS:
BMC40941I              AUX = NO              (INCLUDE XML/LOB/ALL/HISTORY AUXILIARY

BMC40931I                  OPTIONS:
BMC40941I              USEHDROBIDS = YES          (USE HEADER PAGE IDS FOR OBIDXLAT)
BMC40931I                  OPTIONS:
BMC40941I              AUX = NO              (INCLUDE XML/LOB/ALL/HISTORY AUXILIARY

BMC40931I                  OPTIONS:
BMC40941I              USEHDROBIDS = YES          (USE HEADER PAGE IDS FOR OBIDXLAT)
BMC40931I                  OPTIONS:
BMC40941I              AUX = NO              (INCLUDE XML/LOB/ALL/HISTORY AUXILIARY
```
Chapter 4  Building and executing RECOVER PLUS jobs  351

Using MSGLEVEL to control RECOVER PLUS output
Using MSGLEVEL to control RECOVER PLUS output

Figure 58: AFRPR001 with MSGLEVEL(0)

$BMCPRM$ : BMSG,EQUALS
PARMLIST :
SORT FIELDS=(30,20,BI,A),DYNALLOC,FILSZ=E797
RECORD TYPE=V,LENGTH=(8192,8192,8192,32,1000)
OPTION SORTDD=L001,MSGDDN=LOGOU001,AVGRLEN=1000
WER161B  ALTERNATE PARM USED
WER164B  2,260K BYTES OF VIRTUAL STORAGE AVAILABLE, MAX REQUESTED.
WER164B  44K BYTES RESERVE REQUESTED, 2,212K BYTES USED
WER410B  1,956K BYTES OF VIRTUAL STORAGE AVAILABLE ABOVE THE 16MEG LINE.
WER410B  0 BYTES RESERVE REQUESTED, 1,956K BYTES USED
WER036B  G=150,B=13672,SEGLEN=4200,BIAS=25
WER162B  0 PREALLOCATED SORTWORK TRACKS, 960 DYNAMICALLY ALLOCATED,
WER162B  0 ACQUIRED IN SECONDARY EXTENTS, 0 RELEASED, TOTAL OF 43 TRACKS USED
WER045C  END SORT PHASE
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 56 on page 349)
- AFRSTMT, which is similar to the output with MSGLEVEL(0) (Figure 57 on page 350)
- AFRPRINT (Figure 59 on page 353)
- AFRPR001 (Figure 60 on page 353)
- AFROSUM (Figure 61 on page 355)

Figure 59: AFRPRINT with MSGLEVEL(1)

```plaintext
BMC40018I  RECOVER PLUS FOR DB2 V11.1.00  (RECOVER AFRPRINT)
BMC40473I  COPYRIGHT BMC SOFTWARE INC. 1991-2013
BMC96173I  RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884 7,769,718 AND 8,161,001
BMC40474I  CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
BMC96400I  LOG INPUT PHASE MESSAGES ARE IN FILE AFRPR001
BMC96400I  MERGE PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR001
BMC40301I  TABLESPACE = BMCDBSMP.BMCTS014
BMC96400I  BUILD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR001
BMC40303I  INDEX = RMD.BMCIX014
```

Figure 60: AFRPR001 with MSGLEVEL(1)

```plaintext
BMC40018I  RECOVER PLUS FOR DB2 V11.1.00  (RECOVER AFRPRINT)
BMC40473I  COPYRIGHT BMC SOFTWARE INC. 1991-2013
BMC96173I  RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884 7,769,718 AND 8,161,001
BMC40474I  CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
BMC40016I  LOG INPUT PHASE STARTING 08/28/2012 10:46:46
BMC40954I  AVAILABLE MEMORY BELOW THE LINE = 10044K, TOTAL AVAILABLE MEMORY (AT LEAST) = 1370496K
```
BMC40441I  BMCSORT PRODUCT LOCATED AND WILL BE USED
BMC40020I  SYSEVENT STGTEST STORAGE AVAILABLE = 5606844
BMC40020I  SORT FILE SIZE = 2448384, SORT MEMORY = 242407
BMC40725I  PROCESSING LOG RANGE:
BMC40302I    DSN = DEFQCAT.LOGCOPY1.DS11.DATA
BMC40555I    FROMRBA = X'00000000005917EE13000'  TORBA = X'00000000005917F068756'
BMC40193I  3 LOG DATASET BUFFERS OF 737280 BYTES EACH WERE ALLOCATED
BMC40302I    DSN = DEFQCAT.LOGCOPY1.DS11.DATA
BMC40117I  **********     PERFORMANCE STATISTICS FROM LOG READ STEP     **********
BMC40120I      1 FILES READ     0 I/O WAITS        I/O WAIT TIME = 00:00:00
BMC40175I      2 WAITS FOR THREAD    WAIT TIME = 00:00:00
BMC40124I      3 SORT WAITS      SORT WAIT TIME = 00:00:00
BMC40172I  24995 LOG RECORDS AND 1759648 BYTES OF LOG DATA PROCESSED
BMC40302I    DSN = DEFQCAT.LOGCOPY1.DS11.DATA
BMC40173I      24995 LOG RECORDS AND 1759648 BYTES OF LOG DATA PROCESSED
BMC40012I  LOG INPUT PHASE COMPLETE. ELAPSED TIME = 00:00:00, TIME SINCE UTILITY START =
BMC40868I  LOG INPUT PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00
BMC40016I  MERGE PHASE STARTING 08/28/2012 10:46:46
BMC40301I    TABLESPACE = BMCDBSMP.BMCTS014
BMC40793I    EXTRACTING KEYS FOR:
BMC40303I         INDEX = RMD.BMCIX014
BMC96304I  THIS PHASE WILL RUN IN EXECUTION SUBTASK 1
BMC40102I  MERGE TABLESPACE STEP STARTING
BMC40795I  FULL COPY DATASET IS AFR.BMCDBSMP.BMCTS014.IC.G0003V00
BMC40799I  MERGE STARTING FOR DATA SET DEFQCAT.DSNDDB.BMCDBSMP.BMCTS014.10001.A001
BMC40088I  WAIT TIME FOR SORT OUTPUT PHASE = 00:00:00
BMC40906I  NUMBER OF PAGES EXAMINED = 4999, ROWS EXAMINED = 4999, KEYS EXTRACTED = 4999
BMC40907I  INDIRECT REFERENCES: FORWARD = 0, BACKWARD = 0
BMC40981I  FULL IC AFR.BMCDBSMP.BMCTS014.IC.G0003V00 READ WAITS = 3, WAIT TIME = 00:00:00
BMC40979I  LOG INPUT WAITS = 1, WAIT TIME = 00:00:00
BMC40905I  SPACE WRITE WAITS = 14, WAIT TIME = 00:00:00
BMC40980I  KEY OUTPUT WAITS = 0, WAIT TIME = 00:00:00
BMC40985I  NUMBER OF PAGES OUTPUT IS = 5003
BMC40981I  NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 5003, USED = 5003
BMC40791I  NUMBER OF PAGES READ FROM FULL COPY TOTAL = 5003, USED = 5003
BMC40792I  LOG RECORDS THIS PAGESET: FROM SORT = 24995 (1759K BYTES), SELECTED = 24995
BMC40981I  LOG RECORDS TOTAL: FROM SORT = 24995 (1759K BYTES), SELECTED = 24995
BMC40985I  NUMBER OF PAGES OUTPUT IS = 5003
BMC40901I  MERGE TABLESPACE STEP COMPLETE
BMC40012I  MERGE PHASE COMPLETE. ELAPSED TIME = 00:00:01, TIME SINCE UTILITY START =
BMC40868I  MERGE PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00
BMC40016I  BUILD PHASE STARTING 08/28/2012 10:46:48
BMC40300I    TABLESPACE = BMCDBSMP.BMCTS014
BMC40793I    EXTRACTING KEYS FOR:
BMC40303I         INDEX = RMD.BMCIX014
BMC40484I END OF PHASE SUBTASK MESSAGES
THIS PHASE WILL RUN IN EXECUTION SUBTASK 1

BUFFER CREATED FOR PAGE SIZE 4K, REFRESH=1

TOTAL WAITS IN BUFFER MANAGER = 1

RANDOM WRITE WAITS = 0, RANDOM READ WAITS = 0, RANDOM SYNC WAITS = 0

CYLINDER待ち = 1, SEQUENTIAL SYNC待ち = 0

CYLINDERS WAIT = 00:00:00, RANDOM WRITE TIME = 00:00:00, RANDOM READ TIME = 00:00:00

Cylinder Wait Time = 00:00:00

BUILD STATISTICS: # INPUT RECORDS = 4999 # KEY VALUES = 4999

# WAITs FOR SORT = 0

GETPAGE CALLS = 52 # PAGES WRITTEN = 20 # LEAF PAGES = 16

WAIT TIME FOR BUFFER MANAGER = 00:00:00

TOTAL TIME FOR WAITS = 00:00:00, WAIT TIME FOR SORT = 00:00:00

BUILD PHASE COMPLETE. ELAPSED TIME = 00:00:00, TIME SINCE UTILITY START = 00:00:04

BUILD PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00

1 SYNCSORT FOR Z/OS 2.3.1.0NI U.S. PATENTS: 4210961, 5117495 (C) 2007 SYNCSORT INC. DATE=2012/241 TIME=10.46.46

BMCSORT ENGINE - IF PROBLEMS, CONTACT BMC SUPPORT AT 1-800-537-1813

Figure 61: AFROSUM with MSGLEVEL(1)
Using MSGLEVEL to control RECOVER PLUS output

When you specify MSGLEVEL(2), RECOVER PLUS produces the following output files:

- AFRSUMMARY, which is identical to the output of MSGLEVEL(0) (Figure 56 on page 349)
- AFRSTMT (Figure 62 on page 357)
- AFRPRINT and AFRPR001 (Figure 63 on page 360 and Figure 64 on page 360)
- **AFROSUM**, which is identical to the output of MSGLEVEL(1) (Figure 61 on page 355)

- **AFRPLAN** (Figure 65 on page 364)

**Figure 62: AFRSTMT with MSGLEVEL(2)**

```
BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC40471I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40931I PARM LIST:   DB2 SSID = DEFQ
BMC40932I                UTILID = EXAMPML2
BMC40009I UTILITY NEW/RESTART REQUESTED - SYNCPOINT RESTART WILL OCCUR IF UTILID IS IN
BMCUTIL
BMC40933I                RESTART = NEW/RESTART
BMC40934I                MSGLEVEL = 2
BMC40930I        PARM NOT CODED - CHECKPT
BMC40930I        PARM NOT CODED - RDB2STAT
BMC40930I        PARM NOT CODED - AFROPTS
BMC40930I        PARM NOT CODED - ACAOPTS
BMC40924I              PLANRECV = AFRB111T       (MAIN PLAN)
BMC40241I            PUBLICPLAN = YES            (GRANT PLAN TO PUBLIC)
BMC40547I              INDEXLOG = NO             (DEFAULT USE OF INDEXLOG STRATEGY)
BMC40941I                IXRECP = NO             (SET INDEX RECOVER PENDING)
BMC40938I                WKUNIT = SYSALLDA       (WORK UNIT NAME)
BMC40241I            PUBLICPLAN = YES            (GRANT PLAN TO PUBLIC)
BMC40944I              OPNDB2ID = YES            (ACQUIRE RACF AUTHORITY OF DB2)
BMC40927I              RDB2STAT = YES            (RESET DB2 OBJECT STATUS)
BMC40930I        PARM NOT CODED - CHECKPT
BMC40930I        PARM NOT CODED - RDB2STAT
BMC40930I        PARM NOT CODED - AFROPTS
BMC40930I        PARM NOT CODED - ACAOPTS
BMC40963I               SORTNUM = SORT DEFAULT   (WORK DATASETS FOR LOG AND INDEX SORTS)
BMC40926I               CHECKPT = PHASE          (CHECKPOINTING)
BMC40945I              CHECKINT = 0              (CHECKPOINT INTERVAL MINUTES)
BMC40537I            ERRCONT = 10             (DEFAULT MAX SEVERE ERRORS)
BMC40956I            RCLTSK = 10              (MAXIMUM RECALL TASKS)
BMC40925I              TBUFFS = 1000           (BUFFER MANAGER PAGES)
BMC40964I            AUTOSIZE = YES            (AUTO SIZE OUTPUT COPIES AND OUTPUT
ACCUM FILES)
BMC409631            EATTR = NONE            (EXTENDED ATTRIBUTES FOR DYNAMIC
ALLOCATION OF OUTPUT COPIES)
BMC409351           OUTCOPY = ASCODED        (OUTCOPY CREATION DIRECTIVE)
BMC40893I               RESINV = (    0K)       (MEMORY BELOW 16MB EXCLUDED FROM USE BY
SORT)
BMC409621            SORTDEV = SORT DEFAULT   (DEVICE TYPE FOR LOG AND INDEX SORTS)
BMC409631            SORTNUM = SORT DEFAULT   (WORK DATASETS FOR LOG AND INDEX SORTS)
BMC409631            KSORTSHARE = YES            (KEY Sorts Shared By All Tasks)
BMC409421            MAXSORT = 10             (DEFAULT)  (MAX CONCURRENT INDEX KEY SORTS)
BMC409612            MAXLSORT = 50            (MAX CONCURRENT LOG SORTS)
BMC408911            MAXDRIVE = 0            (DEFAULT MAX TAPE DRIVES)
BMC408921            MAXLOGS = 3            (DEFAULT MAX CONCURRENT LOG FILES)
BMC409631            EATTR = NONE            (EXTENDED ATTRIBUTES FOR DYNAMIC
ALLOCATION OF OUTPUT COPIES)
BMC409351           OUTCOPY = ASCODED        (OUTCOPY CREATION DIRECTIVE)
BMC40893I               RESINV = (    0K)       (MEMORY BELOW 16MB EXCLUDED FROM USE BY
SORT)
BMC409621            SORTDEV = SORT DEFAULT   (DEVICE TYPE FOR LOG AND INDEX SORTS)
BMC409631            SORTNUM = SORT DEFAULT   (WORK DATASETS FOR LOG AND INDEX SORTS)
BMC409631            KSORTSHARE = YES            (KEY Sorts Shared By All Tasks)
BMC409421            MAXSORT = 10             (DEFAULT)  (MAX CONCURRENT INDEX KEY SORTS)
BMC409612            MAXLSORT = 50            (MAX CONCURRENT LOG SORTS)
BMC408911            MAXDRIVE = 0            (DEFAULT MAX TAPE DRIVES)
BMC408921            MAXLOGS = 3            (DEFAULT MAX CONCURRENT LOG FILES)
```

**Chapter 4  Building and executing RECOVER PLUS jobs** 357
BMC96251I  DISKIORATE = 100  (DISK READ RATE IN MB/SEC)
BMC96252I  CPUMIPS = 200  (CPU PROCESSING SPEED IN MIPS)
BMC96035I  ----- DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE
BMC96113I  COPIES = (FC,LP,LB)  (DEFAULT SEQUENCE FOR IMAGE COPIES)
BMC96114I  ACCUMS = (LP,LB)  (DEFAULT SEQUENCE FOR CA FILES)
BMC96112I  LOGS = (ACT1,ACT2)  (DEFAULT SEQUENCE FOR LOG FILES)
BMC96112I

BMC96035I  ----- DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE
BMC96113I  COPIES = (RP,RB,FC)  (DEFAULT SEQUENCE FOR IMAGE COPIES)
BMC96114I  ACCUMS = (RP,RB)  (DEFAULT SEQUENCE FOR CA FILES)
BMC96112I  LOGS = (ACT1,ACT2)  (DEFAULT SEQUENCE FOR LOG FILES)
BMC96112I

BMC96035I  ----- TAPE DEVICES
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 = CARTVTS
BMC40967I  TAPE DEVICE08 = CR8NS
BMC40967I  TAPE DEVICE09 = CRT9NS
BMC40967I  TAPE DEVICE10 = DANBARR
BMC40967I  TAPE DEVICE11 = EXT3592
BMC40967I  TAPE DEVICE12 = IBMATL
BMC40967I  TAPE DEVICE13 = IBMVTS
BMC40967I  TAPE DEVICE14 = PHXATL
BMC40967I  TAPE DEVICE15 = SA3590
BMC40967I  TAPE DEVICE16 = TAPEABL
BMC40967I  TAPE DEVICE17 = 3590B
BMC40967I  TAPE DEVICE18 = 3590C
BMC40967I  TAPE DEVICE19 = 3480
BMC40967I  TAPE DEVICE20 = 3400-9
BMC40967I  TAPE DEVICE21 = SYS3480R
BMC40967I  TAPE DEVICE22 = SYS348XR

BMC40018I  RECOVER PLUS FOR DB2 V11.2.00
BMC40876I  MAINT:  NO RECOVER PLUS PTFS APPLIED

BMC96010I  SOLUTION COMMON CODE V11.2.01
BMC40876I  MAINT:  NO SOLUTION COMMON CODE PTFS APPLIED

BMC96010I  BMCSORT ENGINE V2.3.01
BMC40876I  MAINT:  NO BMCSORT ENGINE PTFS APPLIED

BMC96010I  PACKAGE CHECKING AND BINDING STARTING 08/28/2014 10:47:23
BMC40287I  ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00

BMC40020I  EBCDIC CCSID = 37
BMC96014I  DB2 VERSION = 1010  SITE TYPE = LOCAL  MODE = NFM
BMC40954I  AVAILABLE MEMORY BELOW THE LINE = 10068K, TOTAL AVAILABLE MEMORY (AT LEAST) = 1373588K
BMC40037I  UTILID EXAMPML2 WAS NOT FOUND IN THE BMCUTIL TABLE - UTILITY WILL RUN AS IF 'NEW' WERE SPECIFIED

BMC40611I  INPUT STATEMENTS:
BMC40101I  OPTION EARLYRECALL INDEXLOG NO
BMC40101I  RECOVER TABLESPACE BMCDBSMP.BMCTS014
BMC40101I  REBUILD INDEX (RMD.BMCIX014)
BMC40101I  INDEXLOG STRATEGY DISABLED
BMC40960I  ZIIP ENABLED (0) USING SUBSYSTEM ACP1
BMC40937I  z/OS RELEASE = 1.13.00, PID = HBB7780, SMS RELEASE = 1.13.0
Chapter 4 Building and executing RECOVER PLUS jobs

Using MSGLEVEL to control RECOVER PLUS output

BMC40157I AVAILABLE REGION BELOW 16M = 10056K, AVAILABLE REGION ABOVE 16M = 1362336K,
NUMBER OF CPUS = 5
BMC40020I BYT:0, 0, 0; BLK:0, 0, 0; TOT:298932, 5608020, 5608020; iter: 1; RC:4

BMC40335I ANALYZE STARTING 08/28/2014 10:47:23

BMC40335I CATALOG LOOKUP STARTING 08/28/2014 10:47:23
BMC40028I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40336I PROC IX STMT FINISHED.
BMC40028I 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.
BMC40028I 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 SYSIX FINISHED.
BMC40028I 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 SYSKY FINISHED.
BMC40028I 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 SYSFL FINISHED.
BMC40028I 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 SYSVOL FINISHED.
BMC40028I 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.
BMC40028I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40633I USING STANDARD DB2 SECURITY

BMC40336I DB LOOKUP FINISHED.
BMC40028I 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.
BMC40028I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40336I IP LOOKUP FINISHED.
BMC40028I 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.
BMC40028I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40705I VOLUMES IN STOGROUP SYSDEFLT WILL BE USED IN THE FOLLOWING ORDER:
BMC40707I       (DEVXXX)

BMC40336I SCAN SYSTB FINISHED.
BMC40028I 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.
Figure 63: AFRPRINT with MSGLEVEL(2)

Figure 64: AFRPR001 with MSGLEVEL(2)
Chapter 4  Building and executing RECOVER PLUS jobs  361

8.161.001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40016I LOG INPUT PHASE STARTING 08/28/2014 10:47:25
BMC40954I AVAILABLE MEMORY BELOW THE LINE = 10040K, TOTAL AVAILABLE MEMORY (AT LEAST) = 1370484K

BMC40441I BMCSORT PRODUCT LOCATED AND WILL BE USED
BMC40020I ORG-BEL:0, 0, 0; EXT:0, 0, 0; TOT:5604768; ITER: 1; RC:4
BMC40020I SYSEVENT STGTEST STORAGE AVAILABLE = 5604768
BMC40020I OPT-SORT:SBEL:524288; ABOV:523763712; HYPR:879087616; REAS:0; MMEM:0; NUMS:2;
SEAD:2BE60E70; RSV1:0;
BMC40020I OPT-SENT:AVGR:1000; FSZ: 797; SBEL:262144; ABOV:418643968; HYPR:226;
DATA:0; KEYL:21; RSV1:80; RSV2:60
BMC40020I SORT FILE SIZE = 2448384, SORT MEMORY = 242407
BMC40725I PROCESSING LOG RANGE:
BMC40302I          DSN = DEFQCAT.LOGCOPY1.DS11.DATA
BMC40555I          FROMRBA  = X'0000000005917EE13000'  TORBA  = X'0000000005917F068756'
BMC40193I 3 LOG DATASET Buffers of 737280 bytes each were allocated
BMC40302I          DSN = DEFQCAT.LOGCOPY1.DS11.DATA
BMC40352I DDNAME SYS00004 (DSN DEFQCAT.LOGCOPY1.DS11.DATA) was successfully unallocated

BMC40117I ********** PERFORMANCE STATISTICS FROM LOG READ STEP **********
BMC40120I      1 FILES READ     0 I/O WAITS        I/O WAIT TIME = 00:00:00
BMC40175I                       2 WAITS FOR THREAD     WAIT TIME = 00:00:00
BMC40124I                       3 SORT WAITS      SORT WAIT TIME = 00:00:00
BMC40172I                       24995 LOG RECORDS AND 1759648 BYTES OF LOG DATA PROCESSED

BMC40120I      1 FILES READ     0 I/O WAITS        I/O WAIT TIME = 00:00:00
BMC40175I                       2 WAITS FOR THREAD     WAIT TIME = 00:00:00
BMC40124I                       3 SORT WAITS      SORT WAIT TIME = 00:00:00
BMC40172I                       24995 LOG RECORDS AND 1759648 BYTES OF LOG DATA PROCESSED

BMC40172I LOG INPUT PHASE COMPLETE. ELAPSED TIME = 00:00:00, TIME SINCE UTILITY START = 00:00:02
BMC400868I LOG INPUT PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00

BMC40848I MESSAGES FROM PHASE SUBTASK:

BMC40350I DATA SET DEFQCAT.LOGCOPY1.DS11.DATA WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00004 ELAPSED = 00:00:00

BMC40485I END OF PHASE SUBTASK MESSAGES

BMC40016I MERGE PHASE STARTING 08/28/2014 10:47:25
BMC40301I          TABLESPACE = BMCDBSMP.BMCTS014
BMC40484I MESSAGES FROM PHASE SUBTASK:

BMC40345I    SPACE = BMCDBSMP.BMCTS014
BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET
DEFCAT.DSNDBC.BMCTS014.I0001.A001
BMC40356I AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET
DEFCAT.DSNDBC.BMCTS014.I0001.A001
BMC40350I DATA SET DEFCAT.DSNDBC.BMCTS014.I0001.A001 WAS SUCCESSFULLY
ALLOCATED TO DDNAME SYS00007 ELAPSED = 00:00:00

BMC40485I END OF PHASE SUBTASK MESSAGES

BMC40793I EXTRACTING KEYS FOR:
BMC40303I          INDEX = RMD.BMCIX014
Using MSGLEVEL to control RECOVER PLUS output

Chapter 4  Building and executing RECOVER PLUS jobs  363
Figure 65: AFRPLAN with MSGLEVEL(2)

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884 7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40700I EXECUTION PLAN SUMMARY:

BMC40701I PHASE: LOG INPUT
BMC40721I PHASE INITIALIZATION:
BMC40738I INITIALIZE LOG RECORD SORT 1
BMC40814I ESTIMATED RECORD SIZE = 1000
BMC40813I ESTIMATED NUMBER OF RECORDS = 797
BMC40738I ALLOCATE 2 POINTER BUFFERS (53092 BYTES EACH)
BMC40702I STEP: READ LOG PAGES
BMC40736I STEP INITIALIZATION:
BMC40738I ALLOCATE INPUT LOG FILE
BMC40302I DSN = DEFQCAT.LOGCOPY1.DS11.DATA
BMC40741I STEP PROCESSING:
BMC40738I FILE OPTIMIZATION
BMC40738I OPEN LOG DATASET
BMC40302I DSN = DEFQCAT.LOGCOPY1.DS11.DATA
BMC40738I ALLOCATE LOG DATASET BUFFER(S)
BMC40302I DSN = DEFQCAT.LOGCOPY1.DS11.DATA
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
BMC40345I SPACE = BMCDBSMP.BMCTS014
BMC40738I ALLOCATE SPACE (READ/WRITE)
BMC40738I CREATE SPACE
BMC40345I SPACE = BMCDBSMP.BMCTS014
BMC40738I ALLOCATE SPACE (READ/WRITE)
BMC40738I ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I FILE OPTIMIZATION
BMC40738I ALLOCATE 2 SPACE OUTPUT BUFFER(S)

Using MSGLEVEL to control RECOVER PLUS output
Chapter 4  Building and executing RECOVER PLUS jobs  365
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. The following table describes the major 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>
</tbody>
</table>
### Multiple table space recovery and index rebuilds

The following figure 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 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 the following figure.

Figure 66: Phases used for table space recovery with indexes rebuilt

RECOVER TABLESPACE tableSpaceName1 OUTCOPY YES . . .

. . . . .
TABLESPACE tableSpaceName2
REBUILD INDEX(ALL) TABLESPACE tableSpaceName1 NOWORKDDN

. . . . .
REBUILD INDEX(ALL) TABLESPACE tableSpaceName3 NOWORKDDN
Parallel index key sorts and multitasking index rebuilds

The following figure 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 the following figure.

Figure 67: Phases used for table space recovery and parallel index sorts with multitasking index rebuilds

```
OPTIONS MAXKSORT 2
RECOVER TABLESPACE tableSpaceName1

. . . . .

REBUILD INDEX(ALL) TABLESPACE tableSpaceName1
```
Multiple log sorts, MERGE phases, and parallel index key sorts with multitasking index rebuilds

The following figure 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 369, 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.
Examples of RECOVER PLUS operations

Figure 68: Phases used for table space recovery with multiple log sorts and parallel index sorts with multitasking index rebuilds

Options MAXSORT 2 MAXLSORT 2 KSORTSHARE NO
RECOVER TABLESPACE tableSpaceName1

TABLESPACE tableSpaceName1
REBUILD INDEX(ALL) TABLESPACE tableSpaceName1
REBUILD INDEX(ALL) TABLESPACE tableSpaceName1

372 RECOVER PLUS for DB2 Reference Manual
Multiple table space and index recoveries

The following figure 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 the following figure.

Figure 69: Phases used for table space and index space recovery

RECOVER TABLESPACE tableSpaceName1

TABLESPACE tableSpaceName
RECOVER INDEXSPACE(ALL) TABLESPACE tableSpaceName1

RECOVER INDEXSPACE(ALL) TABLESPACE tableSpaceName

Examples of RECOVER PLUS operations
BACKOUT recovery

The following figure 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 70: Phases used for BACKOUT recovery

```
OPTIONS BACKOUT YES INDEXLOG YES
RECOVER TABLESPACE tableSpaceName1 TOLOGPOINT LASTQUIESCE
RECOVER INDEXSPACE (indexName1)
```
**Instant Snapshot recovery**

The following figure 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.

**Figure 71: Phases used for table space and index space recovery for Instant Snapshots**

```
OPTIONS XBMID DBXM
RECOVER TABLESPACE tableSpaceName1

TABLESPACE tableSpaceNamen
RECOVER INDEXSPACE(ALL) TABLESPACE tableSpaceName1

RECOVER INDEXSPACE(ALL) TABLESPACE tableSpaceNamen
```
Examples of RECOVER PLUS jobs

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 325
- “DD statements common to all RECOVER PLUS executions” on page 326
- “Using MSGLEVEL to control RECOVER PLUS output” on page 348

See “RECOVER PLUS syntax” on page 87 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).

The following tables provide 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 15: Job examples for Utility parameters**

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARM VALUE</td>
<td>all examples</td>
</tr>
<tr>
<td>REGION</td>
<td>all examples</td>
</tr>
<tr>
<td>Command or keyword</td>
<td>Relevant examples</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>NEW/RESTART</td>
<td>“Example 1: Previewing the recovery plan” on page 384</td>
</tr>
<tr>
<td></td>
<td>“Example 2B: Using RECOVER INDEX” on page 394</td>
</tr>
<tr>
<td></td>
<td>“Example 3: Creating and dynamically allocating output copies” on page 410 through “Example 9: Recovering to a non-DB2 data set” on page 417</td>
</tr>
<tr>
<td></td>
<td>“Example 10B: Overriding the IXRECP installation option” on page 429</td>
</tr>
<tr>
<td></td>
<td>“Example 11: Recovering using a change accumulation file” on page 436</td>
</tr>
<tr>
<td></td>
<td>“Example 13: Using the LOGSCAN command” on page 437 through “Example 17: Using timestamp recovery” on page 450</td>
</tr>
<tr>
<td>NEW</td>
<td>“Example 2A: Using REBUILD INDEX” on page 389</td>
</tr>
<tr>
<td></td>
<td>“Example 10A: Overriding several installation options” on page 429</td>
</tr>
<tr>
<td>MSGLEVEL</td>
<td>“Example 2: Recovering to the current state” on page 389 through “Example 9: Recovering to a non-DB2 data set” on page 417</td>
</tr>
<tr>
<td></td>
<td>“Example 10B: Overriding the IXRECP installation option” on page 429 through “Example 17: Using timestamp recovery” on page 450</td>
</tr>
<tr>
<td>MAINT</td>
<td>“Example 12: Using the MAINT parameter” on page 437</td>
</tr>
</tbody>
</table>

Table 16: Job examples using the OPTIONS command

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTIONS</td>
<td>“Example 2: Recovering to the current state” on page 389</td>
</tr>
<tr>
<td></td>
<td>“Example 5: Recovering to a specific log point” on page 413</td>
</tr>
<tr>
<td>BACKOUT</td>
<td>“Example 17: Using timestamp recovery” on page 450</td>
</tr>
<tr>
<td>EARLYRECALL</td>
<td>“Example 2: Recovering to the current state” on page 389</td>
</tr>
<tr>
<td>Command or keyword</td>
<td>Relevant examples</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------</td>
</tr>
</tbody>
</table>
| INDEXLOG          | “Example 2: Recovering to the current state” on page 389  
|                   | “Example 5: Recovering to a specific log point” on page 413  
|                   | “Example 13: Using the LOGSCAN command” on page 437  
| IXRECP            | “Example 10: Overriding installation options” on page 428  
| MAXKSORT          | “Example 15: Using MAXKSORT for parallel index rebuilds” on page 441  
|                   | “Example 16: Using MAXKSORT and recovering a table space” on page 450  
| MAXLSORT          | “Example 18: Using the MAXLSORT default value” on page 451  
| RECOVERYPOINT     | “Example 17: Using timestamp recovery” on page 450  
| RESOURCE SELECTION| “Example 10: Overriding installation options” on page 428  
| SIMULATE YES      | “Example 14: Simulating recovery” on page 440  
| SORTDEVT          | “Example 10: Overriding installation options” on page 428  
| SORTNUM           | “Example 10: Overriding installation options” on page 428  
| USEACCUM          | “Example 11: Recovering using a change accumulation file” on page 436  
| TIMESTAMP         | “Example 17: Using timestamp recovery” on page 450  

Table 17: Job examples using the OUTPUT command

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
</table>
| OUTPUT            | “Example 3: Creating and dynamically allocating output copies” on page 410  
|                   | “Example 17: Using timestamp recovery” on page 450  
| DSNAME            | “Example 3: Creating and dynamically allocating output copies” on page 410  
<p>|                   | “Example 17: Using timestamp recovery” on page 450  |</p>
<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODELDCB</td>
<td>“Example 3: Creating and dynamically allocating output copies” on page 410</td>
</tr>
<tr>
<td>SPACE</td>
<td>“Example 3: Creating and dynamically allocating output copies” on page 410</td>
</tr>
<tr>
<td>UNIT</td>
<td>“Example 3: Creating and dynamically allocating output copies” on page 410</td>
</tr>
<tr>
<td></td>
<td>“Example 17: Using timestamp recovery” on page 450</td>
</tr>
</tbody>
</table>

**Table 18: Job examples using the RECOVER command**

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER</td>
<td>“Example 1: Previewing the recovery plan” on page 384 through “Example 6: Extracting nonpartitioned index keys” on page 415</td>
</tr>
<tr>
<td></td>
<td>“Example 8: Recovering using tape-stacked data sets” on page 416 through “Example 11: Recovering using a change accumulation file” on page 436</td>
</tr>
<tr>
<td></td>
<td>“Example 16: Using MAXSORT and recovering a table space” on page 450 through “Example 17: Using timestamp recovery” on page 450</td>
</tr>
<tr>
<td>ANALYZE</td>
<td>“Example 1: Previewing the recovery plan” on page 384</td>
</tr>
<tr>
<td>BACKOUT</td>
<td>“Example 5: Recovering to a specific log point” on page 413</td>
</tr>
<tr>
<td>DSNUM</td>
<td>“Example 6: Extracting nonpartitioned index keys” on page 415</td>
</tr>
<tr>
<td>INDEP OUTSPACE</td>
<td>“Example 9: Recovering to a non-DB2 data set” on page 417</td>
</tr>
<tr>
<td>INDEX</td>
<td>“Example 5: Recovering to a specific log point” on page 413</td>
</tr>
<tr>
<td></td>
<td>“Example 17: Using timestamp recovery” on page 450</td>
</tr>
<tr>
<td>LASTCOPY</td>
<td>“Example 4: Recovering to a specified copy” on page 410</td>
</tr>
<tr>
<td></td>
<td>“Example 8: Recovering using tape-stacked data sets” on page 416</td>
</tr>
<tr>
<td></td>
<td>“Example 10: Overriding installation options” on page 428</td>
</tr>
<tr>
<td>LASTQUIESCE</td>
<td>“Example 5: Recovering to a specific log point” on page 413</td>
</tr>
<tr>
<td>Command or keyword</td>
<td>Relevant examples</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OUTCOPY</td>
<td>“Example 3: Creating and dynamically allocating output copies” on page 410</td>
</tr>
<tr>
<td>OUTCOPYDDN</td>
<td>“Example 3: Creating and dynamically allocating output copies” on page 410</td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>“Example 1: Previewing the recovery plan” on page 384 through “Example 6: Extracting nonpartitioned index keys” on page 415</td>
</tr>
<tr>
<td></td>
<td>“Example 8: Recovering using tape-stacked data sets” on page 416 through “Example 11: Recovering using a change accumulation file” on page 436</td>
</tr>
<tr>
<td></td>
<td>“Example 16: Using MAXKSORT and recovering a table space” on page 450</td>
</tr>
<tr>
<td>TOCOPY</td>
<td>“Example 4: Recovering to a specified copy” on page 410</td>
</tr>
<tr>
<td></td>
<td>“Example 8: Recovering using tape-stacked data sets” on page 416</td>
</tr>
<tr>
<td></td>
<td>“Example 10: Overriding installation options” on page 428</td>
</tr>
<tr>
<td>TOLOGPOINT</td>
<td>“Example 5: Recovering to a specific log point” on page 413</td>
</tr>
</tbody>
</table>

Table 19: Job examples using the REBUILD command

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>REBUILD</td>
<td>“Example 2: Recovering to the current state” on page 389</td>
</tr>
<tr>
<td></td>
<td>“Example 4: Recovering to a specified copy” on page 410</td>
</tr>
<tr>
<td></td>
<td>“Example 6: Extracting nonpartitioned index keys” on page 415</td>
</tr>
<tr>
<td></td>
<td>“Example 9: Recovering to a non-DB2 data set” on page 417</td>
</tr>
<tr>
<td></td>
<td>“Example 11: Recovering using a change accumulation file” on page 436</td>
</tr>
<tr>
<td></td>
<td>“Example 15: Using MAXKSORT for parallel index rebuilds” on page 441</td>
</tr>
<tr>
<td></td>
<td>“Example 16: Using MAXKSORT and recovering a table space” on page 450</td>
</tr>
<tr>
<td>INDEP OUTSPACE</td>
<td>“Example 9: Recovering to a non-DB2 data set” on page 417</td>
</tr>
</tbody>
</table>
Table 20: Job examples using the RECOVER UNLOADKEYS command

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDEX</td>
<td>“Example 2: Recovering to the current state” on page 389</td>
</tr>
<tr>
<td></td>
<td>“Example 4: Recovering to a specified copy” on page 410</td>
</tr>
<tr>
<td></td>
<td>“Example 6: Extracting nonpartitioned index keys” on page 415</td>
</tr>
<tr>
<td></td>
<td>“Example 9: Recovering to a non-DB2 data set” on page 417</td>
</tr>
<tr>
<td></td>
<td>“Example 11: Recovering using a change accumulation file” on page 436</td>
</tr>
<tr>
<td></td>
<td>“Example 15: Using MAXKSORT for parallel index rebuilds” on page 441</td>
</tr>
<tr>
<td></td>
<td>“Example 16: Using MAXKSORT and recovering a table space” on page 450</td>
</tr>
<tr>
<td>NOWORKDDN</td>
<td>“Example 2: Recovering to the current state” on page 389</td>
</tr>
<tr>
<td></td>
<td>“Example 4: Recovering to a specified copy” on page 410</td>
</tr>
<tr>
<td></td>
<td>“Example 9: Recovering to a non-DB2 data set” on page 417</td>
</tr>
<tr>
<td></td>
<td>“Example 11: Recovering using a change accumulation file” on page 436</td>
</tr>
<tr>
<td>PART</td>
<td>“Example 6: Extracting nonpartitioned index keys” on page 415</td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>“Example 4: Recovering to a specified copy” on page 410</td>
</tr>
<tr>
<td></td>
<td>“Example 15: Using MAXKSORT for parallel index rebuilds” on page 441</td>
</tr>
<tr>
<td></td>
<td>“Example 16: Using MAXKSORT and recovering a table space” on page 450</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER UNLOADKEYS</td>
<td>“Example 6: Extracting nonpartitioned index keys” on page 415</td>
</tr>
<tr>
<td>PART</td>
<td>“Example 6: Extracting nonpartitioned index keys” on page 415</td>
</tr>
</tbody>
</table>
### Table 21: Job examples using the `RECOVER BUILDINDEX` command

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>RECOVER BUILDINDEX</code></td>
<td>“Example 7: Building a nonpartitioned index” on page 415</td>
</tr>
</tbody>
</table>

### Table 22: Job examples using the `ACCUM` command

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ACCUM</code></td>
<td>“Example 11: Recovering using a change accumulation file” on page 436</td>
</tr>
<tr>
<td><code>GROUP</code></td>
<td>“Example 11: Recovering using a change accumulation file” on page 436</td>
</tr>
</tbody>
</table>

### Table 23: Job examples using the `LOGSCAN` command

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>LOGSCAN</code></td>
<td>“Example 13: Using the LOGSCAN command” on page 437</td>
</tr>
<tr>
<td><code>BACKOUT</code></td>
<td>“Example 13: Using the LOGSCAN command” on page 437</td>
</tr>
<tr>
<td><code>INDEXSPACE</code></td>
<td>“Example 13: Using the LOGSCAN command” on page 437</td>
</tr>
<tr>
<td><code>LASTQUIESCE</code></td>
<td>“Example 13: Using the LOGSCAN command” on page 437</td>
</tr>
<tr>
<td><code>TABLESPACE</code></td>
<td>“Example 13: Using the LOGSCAN command” on page 437</td>
</tr>
<tr>
<td><code>TOLOGPOINT</code></td>
<td>“Example 13: Using the LOGSCAN command” on page 437</td>
</tr>
</tbody>
</table>

### Table 24: Job examples using the `SIMRCVR` command

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SIMRCVR</code></td>
<td>“Example 14: Simulating recovery” on page 440</td>
</tr>
<tr>
<td><code>TABLESPACE</code></td>
<td>“Example 14: Simulating recovery” on page 440</td>
</tr>
</tbody>
</table>

### Table 25: Job examples using the `SIMRBLD` command

<table>
<thead>
<tr>
<th>Command or keyword</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SIMRBLD</code></td>
<td>“Example 14: Simulating recovery” on page 440</td>
</tr>
<tr>
<td><code>INDEX</code></td>
<td>“Example 14: Simulating recovery” on page 440</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 the following figure.

Figure 72: Example 1--JCL using ANALYZE ONLY

```
//AFREX01 JOB (PAFR), 'EXAMPLE 1',
//        CLASS=Q,
//        MSGCLASS=X, NOTIFY=&SYSUID
/*ROUTE XEQ BMCPXLX1
/*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 73 on page 384 through Figure 77 on page 389 show the following output files from this RECOVER PLUS job:

- AFRSUMRY
- AFRSTMT
- AFROSUM
- AFRPLAN
- SYSPICK

See Table 11 on page 326 for a description of each of these files.

Figure 73: Example 1--AFRSUMRY using ANALYZE ONLY

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2013
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884
7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
BMC40001I UTILITY EXECUTION STARTING 08/17/2012 12:58:08

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40876I MAINT: NO RECOVER PLUS PTFS APPLIED
Example 1: Previewing the recovery plan

Figure 74: Example 1--AFRSTMT using ANALYZE ONLY

Chapter 5  Examples of RECOVER PLUS jobs  385
Example 1: Previewing the recovery plan

<table>
<thead>
<tr>
<th>BMC40938I</th>
<th>WKUNIT = SYSALLDA</th>
<th>(WORK UNIT NAME)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC40878I</td>
<td>WTOR = YES</td>
<td>(ISSUE WTOR IF SPACE REMAINS IN STOPP STATUS)</td>
</tr>
<tr>
<td>BM96034I</td>
<td>USEHDROBIDS = YES</td>
<td>(USE HEADER PAGE IDS FOR OBIDXLAT)</td>
</tr>
<tr>
<td>BM96235I</td>
<td>AUX = NO</td>
<td>(INCLUDE XML/LOB/ALL/HISTORY AUXILIARY OBJECTS)</td>
</tr>
<tr>
<td>BM96216I</td>
<td>CHECKPT = PHASE</td>
<td>(CHECKPOINTING)</td>
</tr>
<tr>
<td>BM96215I</td>
<td>CHECKINT = 0</td>
<td>(CHECKPOINT INTERVAL MINUTES)</td>
</tr>
<tr>
<td>BM96231I</td>
<td>ERRCONT = 10</td>
<td>(DEFAULT MAX SEVERE ERRORS)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>RCLTSK = 10</td>
<td>(MAXIMUM RECALL TASKS)</td>
</tr>
<tr>
<td>BM96245I</td>
<td>TBUFFS = 1000</td>
<td>(BUFFER MANAGER PAGES)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>HISTORY = YES</td>
<td>(UPDATE BMCHIST TABLE)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>----- DEFAULT OPTIONS: SORT</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>SMCCORE = (OK, OK)</td>
<td>(SORT CORE VALUES)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>RESINV = (OK)</td>
<td>(MEMORY BELOW 16MB EXCLUDED FROM USE BY SORT)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>EXCLUDED FROM USE BY SORT</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>SORBTDEVT = SORT DEFAULT</td>
<td>(DEVICE TYPE FOR LOG AND INDEX Sorts)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>SORTNUM = SORT DEFAULT</td>
<td>(WORK DATASETS FOR LOG AND INDEX Sorts)</td>
</tr>
<tr>
<td>BM96230I</td>
<td>KSORTSHARE = YES</td>
<td>(KEY SORTS SHARED BY ALL TASKS)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>----- DEFAULT Options: Output Copies</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>OUTCOPY = ASCODED</td>
<td>(OUTCOPY CREATION DIRECTIVE)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>AUTOSIZE = YES</td>
<td>(AUTO SIZE OUTPUT COPIES AND OUTPUT ACCUM FILES)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>EATTR = NONE</td>
<td>(EXTENDED ATTRIBUTES FOR DYNAMIC ALLOCATION OF OUTPUT COPIES)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>----- DEFAULT Options: Estimation</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>DISKIORATE = 100</td>
<td>(DISK READ RATE IN MB/SEC)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>CPUMIPS = 200</td>
<td>(CPU PROCESSING SPEED IN MIPS)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>----- DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>COPIES = (FC,LP,LB)</td>
<td>(DEFAULT SEQUENCE FOR IMAGE COPIES)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>ACCUMS = (LP,LB)</td>
<td>(DEFAULT SEQUENCE FOR CA FILES)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>LOGS = (ACT1,ACT2)</td>
<td>(DEFAULT SEQUENCE FOR LOG FILES)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>----- DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>COPIES = (RP,RB,FC)</td>
<td>(DEFAULT SEQUENCE FOR IMAGE COPIES)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>ACCUMS = (RP,RB)</td>
<td>(DEFAULT SEQUENCE FOR CA FILES)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>LOGS = (ACT1,ACT2)</td>
<td>(DEFAULT SEQUENCE FOR LOG FILES)</td>
</tr>
<tr>
<td>BM96035I</td>
<td>----- TAPE Devices</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>TAPE DEVICE01 = 3590-1</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>TAPE DEVICE02 = 3490</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>TAPE DEVICE03 = 3480X</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>TAPE DEVICE04 = ATLVT3</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>TAPE DEVICE05 = ATL3592</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>TAPE DEVICE06 = CART</td>
<td></td>
</tr>
<tr>
<td>BM96035I</td>
<td>TAPE DEVICE07 = CARTVTS</td>
<td></td>
</tr>
</tbody>
</table>
Example 1: Previewing the recovery plan

Chapter 5  Examples of RECOVER PLUS jobs    387
Figure 75: Example 1--AFROSUM using ANALYZE ONLY

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2013
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884
7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40709I OBJECT SUMMARY:
BMC40711I TABLESPACE = BMCDBSMP.BMCTS001
BMC40712I DBID = X’0267’  PSID = X’0002’
BMC40716I ORIGINAL STATUS = RW
BMC40718I STOGROUP = SYSDEFLT PRIQTY = 800 SECQTY = 400
BMC40710I PAGE SIZE = 4K  SEGSIZE = 4
BMC40723I ENCODING SCHEME = EBCDIC

BMC40820I TABLESPACE RECOVERY
BMC40827I LOG RESOURCE RANGE:
BMC40585I FROM LOGPOINT = X’00CA088B59E63E000000’ TO LOGPOINT = X’00CA08A2FF9049000000’
BMC40840I INPUT COPY / SYSCOPY DATA:
BMC40640I 2012-08-17 11.12.24.2272 RBA/LRSN X’00CA088B59E63E000000’ FULL IMAGE COPY
BMC40302I DSN = RDAPXR.BMCDBSMP.BMCTS001.F.T111223
BMC40641I SHRLEVEL = REFERENCE SITETYPE = LP  DSNUM = 0
BMC40640I 2012-08-17 11.12.26.9237 RBA/LRSN X’00CA08BB559E63E000000’ QUIESCE UTILITY POINT - WRITE(YES)

Figure 76: Example 1--AFRPLAN using ANALYZE ONLY

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2013
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884
7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40755I ANALYSIS PLAN SUMMARY:
BMC40701I PHASE:  MERGE
BMC40721I PHASE INITIALIZATION:
BMC40301I TABLESPACE = BMCDBSMP.BMCTS001
BMC40738I ALLOCATE INPUT IMAGE COPY FILE
BMC40302I DSN = RDAPXR.BMCDBSMP.BMCTS001.F.T111223
BMC40738I FILE OPTIMIZATION
BMC40738I ALLOCATE 2 INPUT FILE BUFFER(S)
BMC40302I DSN = RDAPXR.BMCDBSMP.BMCTS001.F.T111223
BMC40738I OPEN IMAGE COPY FILE
BMC40302I DSN = RDAPXR.BMCDBSMP.BMCTS001.F.T111223
BMC40738I SET RECOVERY PENDING (RECP) FLAG
BMC40738I CREATE SPACE
BMC40345I SPACE = BMCDBSMP.BMCTS001
BMC40738I ALLOCATE SPACE (READ/WRITE)
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 ("Example 2A: Using REBUILD INDEX" on page 389), and the other job uses RECOVER INDEX ("Example 2B: Using RECOVER INDEX" on page 394). 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 78: Example 2A--JCL for a simple RECOVER TABLESPACE and REBUILD INDEX**

```
//AFREX02A JOB (5225),'EXAMPLE 2A',
//         CLASS=Q,
//         MSGCLASS=X,NOTIFY=&SYSUID
/*ROUTE XEQ BMCPLX1
/*JOBPARM SYSAFF=DB2A
//*
//RECOVER EXEC PGM=AFRMAIN,REGION=0M,
//             PARM='DGE,EX02AMSG0,NEW,MSGLEVEL(0)'
//STEPLIB  DD DISP=SHR,DSN=product.libraries
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//         DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN     DD *
OPTIONS EARLYRECALL INDEXLOG NO
RECOVER TABLESPACE BMCDBSMP.BMCTS011
REBUILD INDEX (RMD.BMCIX011) NOWORKDDN
/*
```

Figure 79 on page 390 through Figure 81 on page 394 show the output produced by this JCL, which includes the following files:

- AFRSUMRY
- AFRSTMT
- AFRPR001

See Table 11 on page 326 for a description of each of these files.

**Figure 79: Example 2A--AFRSUMRY for a simple REBUILD INDEX**

```
BMC40018I RECOVER PLUS FOR DB2 V11.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2013
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884
7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
BMC40001I UTILITY EXECUTION STARTING 08/17/2012 12:59:30
BMC40018I RECOVER PLUS FOR DB2 V11.1.00
BMC40876I MAINT:  NO RECOVER PLUS PTFS APPLIED
BMC96010I SOLUTION COMMON CODE V11.1.01
BMC40876I MAINT:  NO SOLUTION COMMON CODE PTFS APPLIED
BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT:  BPJ0195  BPJ0198  BPJ0209  BPJ0250  BPJ0263  BPJ0269
BPJ0279
BMC40876I MAINT:  BPJ0288  BPJ0308  BPJ0343  BPJ0357  BPJ0361  BPJ0374
BPJ0390
BMC40876I MAINT:  BPJ0404  BPJ0473  BPJ0503  BPJ0521
BMC40002I UTILITY ID = EX02AMSG0. DB2 SUBSYSTEM ID = DXW.
BMC40094I ATTEMPTING TO CONNECT TO DB2 SUBSYSTEM DXW, USING PLAN AFRB111T
```
A GROUP ATTACH WAS PERFORMED WITH DXW  
SUCCESSFUL CONNECT TO W4  (RELEASE 910 OF DB2) USING PLAN AFRB111T  
DB2 W4  IS MEMBER DXW4 IN DATA SHARING GROUP DSNDXW  
INPUT STATEMENTS PRINTED IN AFRSTMT  
ANALYZE FINISHED.  
ELAPSED TIME = 00:00:03, UTILITY ELAPSED TIME = 00:00:04  
CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00  
PLANNING FINISHED.  
ELAPSED TIME = 00:00:01, UTILITY ELAPSED TIME = 00:00:05  
CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00  
PLAN EXECUTION FINISHED.  
ELAPSED TIME = 00:00:05, UTILITY ELAPSED TIME = 00:00:10  
CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00  
EXECUTION SUMMARY  
TABLE SPACES / PHASES (PARTITIONS) RECOVERED   :  1 (0)  
LOG RECORDS APPLIED                   :  22496  
INDEXES REBUILT                       :  1  
UTILITY EXECUTION COMPLETE 08/17/2012 12:59:41. RETURN CODE = 0

Figure 80: Example 2A--AFRSTMT for a simple REBUILD INDEX
Example 2: Recovering to the current state
Example 2: Recovering to the current state

BMC40967I TAPE DEVICE11 = EXT3592
BMC40967I TAPE DEVICE12 = IBMATL
BMC40967I TAPE DEVICE13 = IBMVTS
BMC40967I TAPE DEVICE14 = PHXATL
BMC40967I TAPE DEVICE15 = SA3590
BMC40967I TAPE DEVICE16 = TAPAEABL
BMC40967I TAPE DEVICE17 = 3590B
BMC40967I TAPE DEVICE18 = 3590E
BMC40967I TAPE DEVICE19 = 3480
BMC40967I TAPE DEVICE20 = 3400-9
BMC40967I TAPE DEVICE21 = SYS3480R
BMC40967I TAPE DEVICE22 = SYS348XR

BMC40018I RECOVER PLUS FOR DB2 V11.1.00
BMC40876I MAINT: NO RECOVER PLUS PTFS APPLIED
BMC96010I SOLUTION COMMON CODE V11.1.01
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFS APPLIED
BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195  BPJ0198  BPJ0209  BPJ0250  BPJ0263  BPJ0269  BPJ0279
BMC40876I MAINT: BPJ0288  BPJ0308  BPJ0343  BPJ0357  BPJ0361  BPJ0374  BPJ0390
BMC40020I EBCDIC CCSID = 37
BMC96014I DB2 VERSION = 910  SITE TYPE = LOCAL  MODE = NFM

BMC40011I INPUT STATEMENTS:
BMC40101I   OPTION EARLYRECALL INDEXLOG NO
BMC40101I   RECOVER TABLESPACE BMCDBSMP.BMCTS011
BMC40101I

BMC40960I INDEXLOG STRATEGY DISABLED
BMC96020I USE OF REAL-TIME STATISTICS ENABLED FOR REBUILD SORT ESTIMATION
BMC96004I ZIIP ENABLED (0) USING SUBSYSTEM ACP1
BMC40937I z/OS RELEASE = 1.13.00,  PID = HBB7780,  SMS RELEASE = 1.13.0
BMC40157I AVAILABLE REGION BELOW 16M = 10056K, AVAILABLE REGION ABOVE 16M = 1362352K, NUMBER OF CPUS = 5

BMC40335I ANALYZE STARTING 08/17/2012 12:59:31
BMC40335I CATALOG LOOKUP STARTING 08/17/2012 12:59:31
BMC40063I USING STANDARD DB2 SECURITY

BMC40705I VOLUMES IN STOGROUP SYSDEFLT WILL BE USED IN THE FOLLOWING ORDER:
BMC40707I   <SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>
BMC40707I   <SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<SMS VOLUME>,<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 = 00:00:01
BMC40288I   CPU TIME     = 00:00:00, UTILITY CPU TIME     = 00:00:00

BMC40335I SYNCHRONIZATION STARTING 08/17/2012 12:59:32
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

BMC96306I MAXLSORT REDUCED FROM 50 TO 17 DUE TO MEMORY CONSTRAINTS
BMC40335I PLANNING STARTING 08/17/2012 12:59:34
BMC40512I LOG FILE RESOURCES:
Example 2B: Using RECOVER INDEX

In the second job in this example, 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 82: Example 2B--JCL for a simple RECOVER TABLESPACE and RECOVER INDEX

```java
//AFREN260 JOB (PAFR), 'EXAMPLE 2B',
//       CLASS=Q,
//       MSGCLASS=X, NOTIFY=&SYSUID
/*ROUTE XEQ BMCPXL1
/*JOBPARM SYSAFF=DB2A
//*
//RECOVER EXEC PGM=AFRMNEN, REGION=OM,
//       PARM='DGE,EX02B, NEW/RESTART, MSGLEVEL(2)'
//STEPLIB DD DISP=SHR, DSN=product.libraries
//        DD DISP=SHR, DSN=DB2.DSNE.xIT
//        DD DISP=SHR, DSN=DB2.DSNLOAD
//SYSIN DD *
OPTIONS EARLYRECALL INDEXLOG YES
RECOVER TABLESPACE BMCDBSMP.BMCTS011
RECOVER INDEX (RMD.BMCIX011)
/*
```

Figure 83 on page 395 through Figure 89 on page 407 show the following output files produced by this JCL:

- AFRSUMRY
- AFRSTMT
- AFRPRINT
- AFRPR001
- AFRPR002
- AFRSOSUM
- AFRPLAN

See Table 11 on page 326 for a description of each of these files.

Figure 83: Example 2B--AFRSUMRY for a simple RECOVER INDEX
Figure 84: Example 2B--AFRSTMT for a simple RECOVER INDEX

Example 2: Recovering to the current state
Example 2: Recovering to the current state

Chapter 5  Examples of RECOVER PLUS jobs  397
Example 2: Recovering to the current state

BMC40967I TAPE DEVICE07 = CARTVTS
BMC40967I TAPE DEVICE08 = CRT8NS
BMC40967I TAPE DEVICE09 = CRT9NS
BMC40967I TAPE DEVICE10 = DANBARR
BMC40967I TAPE DEVICE11 = EXT3592
BMC40967I TAPE DEVICE12 = IBMATL
BMC40967I TAPE DEVICE13 = IBMVTS
BMC40967I TAPE DEVICE14 = PHXATL
BMC40967I TAPE DEVICE15 = SA3590
BMC40967I TAPE DEVICE16 = TAPEABL
BMC40967I TAPE DEVICE17 = 3590B
BMC40967I TAPE DEVICE18 = 3590E
BMC40967I TAPE DEVICE19 = 3480
BMC40967I TAPE DEVICE20 = 3400-9
BMC40967I TAPE DEVICE21 = SYS3480R
BMC40967I TAPE DEVICE22 = SYS348XR

BMC40018I RECOVER PLUS FOR DB2 V11.1.00
BMC40876I MAINT: NO RECOVER PLUS PTFS APPLIED

BMC96010I SOLUTION COMMON CODE V11.1.01
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFS APPLIED

BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0279
BMC40876I MAINT: BPJ0288 BPJ0308 BPJ0343 BPJ0357 BPJ0361 BPJ0374 BPJ0390
BMC40876I MAINT: BPJ0404 BPJ0473 BPJ0503 BPJ0521

BMC40335I PACKAGE CHECKING AND BINDING STARTING 08/17/2012 13:39:17
BMC40336I PACKAGE CHECKING AND BINDING FINISHED.
BMC40287I ELAPSED TIME = 00:00:01, UTILITY ELAPSED TIME = 00:00:00
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40335I ANALYZE STARTING 08/17/2012 13:39:17
BMC40335I CATALOG LOOKUP STARTING 08/17/2012 13:39:17
BMC40287I ELAPSED TIME = 00:00:00, UTILITY ELAPSED TIME = 00:00:01
BMC40288I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00

BMC40611I INPUT STATEMENTS:
BMC40101I OPTION EARLYRECALL INDEXLOG YES
BMC40101I RECOVER TABLESPACE BMCDBSMP.BMCTS011
BMC40101I RECOVER INDEX (RMD.BMCI011)
BMC40101I
BMC40959I INDEXLOG YES STRATEGY REQUESTED
BMC96020I USE OF REAL-TIME STATISTICS ENABLED FOR REBUILD SORT ESTIMATION
BMC40937I z/OS RELEASE = 1.13.00, PID = HBB7780, SMS RELEASE = 1.13.0
BMC40157I AVAILABLE REGION BELOW 16M = 10056K, AVAILABLE REGION ABOVE 16M = 1362352K, NUMBER OF CPUS = 5
BMC40020I BYT:0, 0, 0; BLK:0, 0, 0; TOT:860944, 5115972, 5115972; iter: 1; RC:4

BMC40335I ANALYZE STARTING 08/17/2012 13:39:17
BMC40335I CATALOG LOOKUP STARTING 08/17/2012 13:39:17
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

398 RECOVER PLUS for DB2 Reference Manual
Example 2: Recovering to the current state

Chapter 5  Examples of RECOVER PLUS jobs  399
Example 2: Recovering to the current state

Figure 85: Example 2B--AFRPRINT for a simple RECOVER INDEX
Figure 86: Example 2B--AFRPR001 for a simple RECOVER INDEX

Example 2: Recovering to the current state

Chapter 5 Examples of RECOVER PLUS jobs
BMC40117I ********** PERFORMANCE STATISTICS FROM LOG READ STEP
**********
BMC40120I 3 FILES READ 1985 I/O WAITS I/O WAIT TIME = 00:00:05
BMC40175I 2 WAITS FOR THREAD WAIT TIME = 00:00:00
BMC401241 0 SORT WAITS SORT WAIT TIME = 00:00:00
BMC401241 1 SORT WAITS SORT WAIT TIME = 00:00:00
BMC40172I 12498 LOG RECORDS AND 819866 BYTES OF LOG DATA PROCESSED
BMC40302I DSN = DSNDXW.DXW4.LOGCOPY1.DS01.DATA
BMC40173I TIME SPENT IN DATASET ALLOCATION = 00:00:00
BMC40174I 1455 I/O WAITS I/O WAIT TIME = 00:00:04
BMC40172I 0 LOG RECORDS AND 0 BYTES OF LOG DATA PROCESSED
BMC40302I DSN = DSNDXW.DXW3.LOGCOPY1.DS02.DATA
BMC40173I TIME SPENT IN DATASET ALLOCATION = 00:00:00
BMC40174I 530 I/O WAITS I/O WAIT TIME = 00:00:00
BMC40172I 12498 LOG RECORDS AND 819866 BYTES OF LOG DATA PROCESSED
BMC40302I DSN = DSNDXW.DXW1.LOGCOPY1.DS02.DATA
BMC40173I TIME SPENT IN DATASET ALLOCATION = 00:00:00
BMC40174I 0 I/O WAITS I/O WAIT TIME = 00:00:00
BMC40172I 0 LOG RECORDS AND 0 BYTES OF LOG DATA PROCESSED
BMC40725I PROCESSING LOG RANGE:
BMC40302I DSN = DSNDXW.DXW2.LOGCOPY1.DS04.DATA
BMC40173I TIME SPENT IN DATASET ALLOCATION = 00:00:00
BMC40174I 6255 I/O WAITS I/O WAIT TIME = 00:00:05
BMC40172I 0 LOG RECORDS AND 0 BYTES OF LOG DATA PROCESSED
BMC40012I LOG INPUT PHASE COMPLETE. ELAPSED TIME = 00:00:13, TIME SINCE UTILITY START = 00:00:18
BMC40868I LOG INPUT PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:01
BMC40484I MESSAGES FROM PHASE SUBTASK:
BMC40350I DATA SET DSNDXW.DXW3.LOGCOPY1.DS02.DATA WAS SUCCESSFULLY UNALLOCATED
BMC40117I ********** PERFORMANCE STATISTICS FROM LOG READ STEP
**********
BMC40120I 1 FILES READ 6255 I/O WAITS I/O WAIT TIME = 00:00:05
BMC40175I 390 WAITS FOR THREAD WAIT TIME = 00:00:00
BMC401241 0 SORT WAITS SORT WAIT TIME = 00:00:00
BMC401241 0 SORT WAITS SORT WAIT TIME = 00:00:00
BMC40172I 0 LOG RECORDS AND 0 BYTES OF LOG DATA PROCESSED
BMC40302I DSN = DSNDXW.DXW2.LOGCOPY1.DS04.DATA
BMC40173I TIME SPENT IN DATASET ALLOCATION = 00:00:00
BMC40174I 6255 I/O WAITS I/O WAIT TIME = 00:00:05
BMC40172I 0 LOG RECORDS AND 0 BYTES OF LOG DATA PROCESSED
BMC40012I LOG INPUT PHASE COMPLETE. ELAPSED TIME = 00:00:13, TIME SINCE UTILITY START = 00:00:18
BMC40868I LOG INPUT PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:01
BMC40484I MESSAGES FROM PHASE SUBTASK:
BMC40350I DATA SET DSNDXW.DXW3.LOGCOPY1.DS02.DATA WAS SUCCESSFULLY

Example 2: Recovering to the current state
Example 2: Recovering to the current state

Chapter 5  Examples of RECOVER PLUS jobs  403
Example 2: Recovering to the current state

Figure 87: Example 2B--AFRPR002 for a simple RECOVER INDEX

BMC40018I RECOVER PLUS FOR DB2 V11.1.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2013
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884
7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
Figure 88: AFROSUM for a simple RECOVER INDEX

**Example 2: Recovering to the current state**
Example 2: Recovering to the current state

BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2013
BMC40709I OBJECT SUMMARY:
BMC40301I TABLESPACE = BMCDBSMP.BMCTS011
BMC40711I DBID = X'0267'  PSID = X'000E'
BMC40712I ORIGINAL STATUS = RW
BMC40716I STOGROUP = SYSDEFLT PRIQTY = 1200 SECQTY = 400
BMC40710I PAGE SIZE = 4K  SEGSIZE = 4
BMC40723I ENCODING SCHEME = EBCDIC

BMC40820I TABLESPACE RECOVERY
BMC40827I LOG RESOURCE RANGE:
BMC40585I FROM LOGPOINT = X'00CA08AC036260000000' TO LOGPOINT =
X'DDCOA8AC3005F5000000'
BMC40727I DISCRETE LOG RANGE(S):
BMC40559I FROMLRSN = X'00CA08AC0434E1000000' TOLRSN =
X'DDCOA8AC4924C000000' MEMBER = DXW3 ID = 2
BMC40555I FROMRBA = X'0000000018753BAB1F0' TORBA =
X'00000000018753C9CA9'

BMC40840I INPUT COPY / SYSCOPY DATA:
BMC40640I 2012-08-17 13.38.31.4241 RBA/LRSN
X'DDCOA8AC306260000000' FULL IMAGE COPY
BMC40365I DSN = RDAPXR.BMCDBSMP.BMCTS011.F.T133831
BMC40564I SHRLEVEL = REFERENCE SITETYPE = LP DSNUM = 0
BMC40640I 2012-08-17 13.38.31.5971 RBA/LRSN
X'DDCOA8AC30686000000' QUIESCE UTILITY POINT - WRITE(YES)
BMC40640I 2012-08-17 13.38.32.4810 RBA/LRSN
X'DDCOA8AC04930E000000' QUIESCE UTILITY POINT - WRITE(YES)

BMC40709I OBJECT SUMMARY:
BMC40346I INDEXSPACE = BMCDBSMP.BMCIX011
BMC40711I DBID = X'0267'  PSID = X'0011'
BMC40712I ORIGINAL STATUS = RW
BMC40303I INDEX = RMD.BMCIX011
BMC40640I PAGE SIZE = 4K
BMC40716I STOGROUP = SYSDEFLT PRIQTY = 60 SECQTY = 40
BMC40832I NON-CLUSTERING INDEX
BMC40833I UNIQUE INDEX
BMC40844I TYPE 2

BMC40843I INDEX RECOVERY
BMC40827I LOG RESOURCE RANGE:
BMC40585I FROM LOGPOINT = X'00CA08AC03CEDC000000' TO LOGPOINT =
X'DDCOA8AC3005F5000000'
BMC40727I DISCRETE LOG RANGE(S):
BMC40559I FROMLRSN = X'00CA08AC03CEDC000000' TOLRSN =
X'DDCOA8AC3005F5000000' MEMBER = DXW3 ID = 2
BMC40555I FROMRBA = X'0000000018753B9F1F0' TORBA =
X'00000000018753C9CA9'

BMC40559I FROMLRSN = X'00CA08AC03CEDC000000' TOLRSN =
X'DDCOA8AC3005F5000000' MEMBER = DXW4 ID = 3
BMC40555I FROMRBA = X'00000000018753C9CA9' TORBA =
X'00000000018753C9CA9'

BMC40559I FROMLRSN = X'00CA08AC03CEDC000000' TOLRSN =
X'DDCOA8AC3005F5000000' MEMBER = DXW5 ID = 5
BMC40555I FROMRBA = X'00000000018753C9CA9' TORBA =
X'00000000018753C9CA9'

406  RECOVER PLUS for DB2 Reference Manual
Example 2: Recovering to the current state

**Figure 89: Example 2B--AFRPLAN for a simple RECOVER INDEX**
Example 2: Recovering to the current state
Example 2: Recovering to the current state

BMC40738I MERGE SPACE
BMC40738I TABLESPACE = BMCDBSMP.BMCTS011
BMC40735I PHASE TERMINATION:
BMC40738I DISCONNECT FROM SPACE
BMC40738I DEALLOCATE SPACE
BMC40345I SPACE = BMCDBSMP.BMCTS011
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 = RDAPXR.BMCDBSMP.BMCTS011.F.T133831
BMC40738I FREE 2 INPUT FILE BUFFER(S)
BMC40738I DEALLOCATE INPUT IMAGE COPY FILE
BMC40302I DSN = RDAPXR.BMCDBSMP.BMCTS011.F.T133831
BMC40738I RESET RECOVERY PENDING (RECP) FLAG
BMC40736I END LOG RECORD SORT
BMC40738I FREE 3 LOG RECORD BUFFER(S)
BMC40738I REGISTER RECOVER TO CURRENT IN SYSCOPY
BMC40738I RECORD RESTART INFORMATION

BMC40701I PHASE: MERGE
BMC40721I PHASE INITIALIZATION:
BMC40303I INDEX = RMD.BMCIX011
BMC40738I ALLOCATE 3 LOG RECORD BUFFER(S) (81920 BYTES EACH)
BMC40738I ALLOCATE INPUT IMAGE COPY FILE
BMC40302I DSN = RDAPXR.BMCDBSMP.BMCIX011.F.T133832
BMC40738I FILE OPTIMIZATION
BMC40738I ALLOCATE 2 INPUT FILE BUFFER(S)
BMC40738I OPEN IMAGE COPY FILE
BMC40302I DSN = RDAPXR.BMCDBSMP.BMCIX011.F.T133832
BMC40738I ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I ALLOCATE SPACEMAP BUFFER (4096 BYTES)
BMC40738I ALLOCATE PASTE BUFFER (4096 BYTES)
BMC40738I SET REBUILD PENDING (RBDP) FLAG
BMC40738I CREATE SPACE
BMC40345I SPACE = BMCDBSMP.BMCIX011
BMC40738I ALLOCATE SPACE (READ/WRITE)
BMC40303I INDEX = RMD.BMCIX011
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
BMC40303I INDEX = RMD.BMCIX011
BMC40735I PHASE TERMINATION:
BMC40738I DISCONNECT FROM SPACE
BMC40738I DEALLOCATE SPACE
BMC40345I SPACE = BMCDBSMP.BMCIX011
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 = RDAPXR.BMCDBSMP.BMCIX011.F.T133832
BMC40738I FREE 2 INPUT FILE BUFFER(S)
BMC40738I DEALLOCATE INPUT IMAGE COPY FILE
BMC40302I DSN = RDAPXR.BMCDBSMP.BMCIX011.F.T133832
Example 3: Creating and dynamically allocating output copies

In the following example, 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.

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 the following figure 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 91: 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, EXAMPLE04A, 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 the following figure 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 92: Example 4B--JCL using TOCOPY LASTCOPY

```plaintext
//AFREX04B JOB (PAFR), 'EXAMPLE 4B',
//         CLASS=Q, NOTIFY=&SYSUID,
//         MSGCLASS=X
//*
//*  *---------------------------------------------------------------------*
```
Example 4: Recovering to a specified copy

```sql
COPY TABLESPACE BMCDBSMP.BMCTS001
   FULL YES COPYDDN (CP00001) RESETMOD YES
   SHRLEVEL REFERENCE
COPY TABLESPACE BMCDBSMP.BMCTS002
   FULL YES COPYDDN (CP00002) RESETMOD YES
   SHRLEVEL REFERENCE
COPY TABLESPACE BMCDBSMP.BMCTS003
   FULL YES COPYDDN (CP00003) RESETMOD YES
   SHRLEVEL REFERENCE
/
*/
```

```sql
UPDATE BMCTB001, BMCTB002, BMCTB003
```

```sql
UPDATE RDASTC.BMCTB001 SET COL2 = 'UPDATE1';
UPDATE RDASTC.BMCTB001 SET COL2 = 'UPDATE2';
```

```sql
UPDATE RDASTC.BMCTB002 SET COL2 = 'UPDATE1';
UPDATE RDASTC.BMCTB002 SET COL2 = 'UPDATE2';
```

```sql
UPDATE RDASTC.BMCTB003 SET COL2 = 'UPDATE1';
UPDATE RDASTC.BMCTB003 SET COL2 = 'UPDATE2';
```
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 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 93: Example 5A--JCL using TOLOGPOINT with RECOVER

```plaintext
//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 uses the BACKOUT feature to recover the table space and its indexes.

Figure 94: Example 5B--JCL using TOLOGPOINT and BACKOUT

```plaintext
//AFREX05B JOB (PAFR),'EXAMPLE 5B',
//         CLASS=Q,NOTIFY=&SYSUID,
//         MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN,REGION=0M,
//             PARM='DGE,EXAMPLE5B,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.BMCTS013
   TOLOGPOINT LASTQUIESCE
   BACKOUT
RECOVER INDEX(ALL) TABLESPACE BMCDBSMP.BMCTS013
/*
```
Example 6: Extracting nonpartitioned index keys

Example 6 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 415 to build an index.

Figure 95: Example 6--JCL recovering a partitioned table space and partitioned index and unloading keys for a nonpartitioned index

Example 7: Building a nonpartitioned index

Example 7 uses the keys that were unloaded and sorted in Example 6 to build an index.

See “Example 6: Extracting nonpartitioned index keys” on page 415.
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 the following figure makes one full image copy of each of three table spaces, registers them, and writes them to the same cartridge tape.

Figure 97: Example 8A--JCL for making stacked image copies

```
//AFREXDBA 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
```
Example 8B: Recovering with stacked image copies

The JCL in the following figure 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.

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 the following figure, RECOVER PLUS assumes it and defaults to the RECOVER TABLESPACE non-DB2 data set for index data input.

**Figure 99: Example 9--JCL for simple table space and index recovery to non-DB2 data sets**

//AFREX10 JOB (PAFR), 'EXAMPLE 9',    
//     CLASS=Q, NOTIFY=&SYSUID,      
//     MSGCLASS=X                    
//*                                 
//RECOVER EXEC PGM=AFRMAIN, REGION=0M,  
//     PARM=''DEBA, EXAMPLE9, 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.BMCTS010 INDEP OUTSPACE 
REBUILD INDEX (RMD.BMCIX010) INDEP OUTSPACE NOWORKDDN */

## 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:

  ```sql
  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 INTABLESPACE 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
```

The following figures show the following output files produced by this JCL:

- AFRSUMRY
- AFRSTMT
- AFRPRINT
- AFRPR001
- AFRPR002
- AFROSUM
**Example 9: Recovering to a non-DB2 data set**

**Figure 100: Example 9--AFRSUMRY for recovering to a non-DB2 data set**

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884
7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40001I UTILITY EXECUTION STARTING 08/20/2014 09:51:30

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40876I MAINT: NO RECOVER PLUS PTFS APPLIED

BMC96010I SOLUTION COMMON CODE V11.2.00
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFS APPLIED

BMC96101I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269
BPJ0279
BMC40876I MAINT: BPJ0288 BPJ0308 BPJ0343 BPJ0357 BPJ0361 BPJ0374
BPJ0390
BMC40876I MAINT: BPJ0404 BPJ0473 BPJ0503 BPJ0521

BMC40002I UTILITY ID = EXAMPLE10B. DB2 SUBSYSTEM ID = DEDL.
BMC40094I ATTEMPTING TO CONNECT TO DB2 SUBSYSTEM DEDL, USING PLAN AFRB111T
BMC40024I SUCCESSFUL CONNECT TO DEDL (RELEASE 910 OF DB2) USING PLAN
AFRB111T

BMC40475I INPUT STATEMENTS PRINTED IN AFRSTMT

BMC40336I ANALYZE 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 PLANNING 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 PLAN EXECUTION 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

BMC40476I EXECUTION SUMMARY
BMC40477I TABLE SPACES / PHASES (PARTITIONS) RECOVERED : 1 (0)
BMC40480I INDEXES REBUILT : 1

BMC40005I UTILITY EXECUTION COMPLETE 08/20/2014 09:51:35, RETURN CODE = 4

**Figure 101: Example 9--AFRPRINT for recovering to a non-DB2 data set**

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884
7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC96400I MERGE PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE
AFRPR001
BMC40301I TABLESPACE = BMCDBSMP.BMCTS010
BMC96400I BUILD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE
Example 9: Recovering to a non-DB2 data set

![Example 9: Recovering to a non-DB2 data set](image)

Chapter 5  Examples of RECOVER PLUS jobs  421
Example 9: Recovering to a non-DB2 data set

Figure 103: Example 9--AFRSTMT for recovering to a non-DB2 data set
**Example 9: Recovering to a non-DB2 data set**

### Recovering Strategy
- **INDEXLOG** = NO (Default Use of INDEXLOG Strategy)
- **IXRECP** = NO (Set Index Recover Pending)
- **BACKOUT** = AUTO (Default Point-in-Time Recovery of DB2)

### Input
- **OPNDB2ID** = YES (Acquire RACF Authority of DB2)
- **RDB2STAT** = YES (Reset DB2 Object Status)
- **AMSCAT** = NO (Use 'CATALOG' in IDCAMS Input)
- **WKUNIT** = SYSALLDA (Work Unit Name)
- **WTOR** = YES (Issue WTOR if Space Remains in STOPP Status)
- **USEHDROBIDS** = YES (Use Header Page IDs for OBIDXLAT)
- **AUX** = NO (Include XML/LOB/ALL/Auxiliary Objects)
- **CHECKPT** = PHASE (Checkpointing)
- **CHECKINT** = 0 (Checkpoint Interval Minutes)
- **ERRCONT** = 10 (Default Max Severe Errors)
- **RCLTSK** = 10 (Maximum Recall Tasks)
- **TBUFFS** = 1000 (Buffer Manager Pages)
- **HISTORY** = YES (Update BMCHIST Table)
- **ACFORTSS** = YES (Use ACF2 or TOP SECRET Security)

### History Auxiliary Objects
- **SORTDEV** = SORT DEFAULT (Device Type for Log and Index Sorts)
- **SORTNUM** = SORT DEFAULT (Work datasets for Log and Index Sorts)
- **KSORTSHARE** = YES (Key sorts shared by all tasks)
- **MAXSORT** = 10 (Default) (Max Concurrent Index Key Sorts)
- **MAXLSORT** = 50 (Max Concurrent Log Sorts)
- **MAXDRIVE** = 0 (Default Max Tape Drives)
- **MAXLOGS** = 3 (Default Max Concurrent Log Files)

### Log Files
- **SNCORE** = (OK, OK) (Sort Core Values)
- **RESINV** = (OK) (Memory below 16MB Excluded from Use by Sort)
- **SORTDEVT** = SORT DEFAULT (Device Type for Log and Index Sorts)
- **SORTNUM** = SORT DEFAULT (Work datasets for Log and Index Sorts)
- **KSORTSHARE** = YES (Key sorts shared by all tasks)
- **MAXSORT** = 10 (Default) (Max Concurrent Index Key Sorts)
- **MAXLSORT** = 50 (Max Concurrent Log Sorts)
- **MAXDRIVE** = 0 (Default Max Tape Drives)
- **MAXLOGS** = 3 (Default Max Concurrent Log Files)

### Output Copies
- **OUTCOPY** = ASCODED (Outcopy Creation Directives)
- **AUTOSIZE** = YES (Auto Size Output Copies)
- **EATTR** = NONE (Extended Attributes for Dynamic Allocation of Output Copies)
- **DISKIORATE** = 100 (Disk Read Rate in MB/SEC)
- **CPU MOPS** = 200 (CPU Processing Speed in MIPS)

### Resource Selection Sequence
- **IMAGE COPIES**
  - Local Site: FC, LP, LB
  - Remote Site: RP, RB, RC

### Default Resource Selection Sequence
- **LOCAL SITE**
  - IMAGE COPIES
  - LOGS: ACT1, ACT2

- **REMOTE SITE**
  - IMAGE COPIES
  - ACCUMS: RP, RB
Example 9: Recovering to a non-DB2 data set

```
FILES)
BMC96112I LOGS = (ACT1,ACT2 (DEFAULT SEQUENCE FOR LOG FILES)
BMC96112I          ARC1,ARC2)
BMC96035I ----- TAPE DEVICES
BMC40967I          TAPE DEVICE01 = 3590-1
BMC40967I          TAPE DEVICE02 = 3490
BMC40967I          TAPE DEVICE03 = 3480X
BMC40967I          TAPE DEVICE04 = ATLVT5
BMC40967I          TAPE DEVICE05 = ATL3592
BMC40967I          TAPE DEVICE06 = CART
BMC40967I          TAPE DEVICE07 = CARTVTS
BMC40967I          TAPE DEVICE08 = CR8NSS
BMC40967I          TAPE DEVICE09 = CRT9NS
BMC40967I          TAPE DEVICE10 = EXT3592
BMC40967I          TAPE DEVICE11 = IBMATL
BMC40967I          TAPE DEVICE12 = IBMVTS
BMC40967I          TAPE DEVICE13 = PHXATL
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

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40876I MAINT:  NO RECOVER PLUS PTFS APPLIED
BMC96010I SOLUTION COMMON CODE V11.1.01
BMC40876I MAINT:  NO SOLUTION COMMON CODE PTFS APPLIED
BMC40876I MAINT:  BPJ0195  BPJ0198  BPJ0209  BPJ0250  BPJ0263  BPJ0269
BMC40876I MAINT:  BPJ0279
BMC40876I MAINT:  BPJ0288  BPJ0308  BPJ0343  BPJ0357  BPJ0361  BPJ0374
BMC40876I MAINT:  BPJ0390
BMC40876I MAINT:  BPJ0404  BPJ0473  BPJ0503  BPJ0521
BMC40335I PACKAGE CHECKING AND BINDING STARTING 08/20/2014 09:51:31
BMC40336I PACKAGE CHECKING AND BINDING 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
BMC40020I EBCDIC CCSID = 37
BMC96014I DB2 VERSION = 910  SITE TYPE = LOCAL  MODE = NFM
BMC40954I AVAILABLE MEMORY BELOW THE LINE = 9044K, TOTAL AVAILABLE MEMORY
AT LEAST) = 1414560K
BMC40611I AVAILABLE REGION BELOW 16M = 9032K, AVAILABLE REGION ABOVE 16M =
1404336K, NUMBER OF CPUS = 5
BMC40028I UTILID EXAMPLE10B ORIGINALLY TERMINATED BEFORE COMPLETING
ANALYZE PHASE. RESTART WILL BE TREATED AS A NEW RUN
BMC40611I INPUT STATEMENTS:
BMC40101I   RECOVER TABLESPACE BMCDBSMP.BMCTS010 INDEP OUTSPACE
BMC40101I   REBUILD INDEX (RMD.BMCIX010) INDEP OUTSPACE NOWORKDDN
BMC40101I
BMC96004I ZIIP ENABLED (0) USING SUBSYSTEM XBSA
BMC40937I z/OS RELEASE = 1.13.00, PID = HBB7780, SMS RELEASE = 1.13.0
BMC40157I AVAILABLE REGION BELOW 16M = 9032K, AVAILABLE REGION ABOVE 16M =
1404336K, NUMBER OF CPUS = 5
BMC40028I UTILID EXAMPLE10B ORIGINALLY TERMINATED BEFORE COMPLETING
ANALYZE PHASE. RESTART WILL BE TREATED AS A NEW RUN
BMC40028I
BMC40028I
BMC40335I ANALYZE STARTING 08/20/2014 09:51:31
BMC40335I CATALOG LOOKUP STARTING 08/20/2014 09:51:31
BMC40336I PROC NO STMT FINISHED.
```
Example 9: Recovering to a non-DB2 data set

Chapter 5  Examples of RECOVER PLUS jobs  425
Example 9: Recovering to a non-DB2 data set

Figure 104: Example 9--AFROSUM for recovering to a non-DB2 data set
Example 9: Recovering to a non-DB2 data set

Figure 105: Example 9--AFRPLAN for recovering to a non-DB2 data set

```
BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884
               7,769,718 AND 8,161,001
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 = BMCDBSMP.BMCTS010
BMC40302I          DSN = RDAPXR.BMCDBSMP.BMCTS010.F.T095108
BMC40738I       ALLOCATE INPUT IMAGE COPY FILE
BMC40738I       FILE OPTIMIZATION
BMC40738I       ALLOCATE 2 INPUT FILE BUFFER(S)
BMC40738I       OPEN IMAGE COPY FILE
BMC40302I          DSN = RDAPXR.BMCDBSMP.BMCTS010.F.T095108
BMC40738I       ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I       ALLOCATE SPACEMAP BUFFER (4096 BYTES)
BMC40738I       ALLOCATE PASTE BUFFER (4096 BYTES)
BMC40738I       CREATE SPACE
BMC40345I          SPACE = BMCDBSMP.BMCTS010
BMC40738I       ALLOCATE SPACE (READ/WRITE)
BMC40738I       ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I       ALLOCATE SPACEMAP BUFFER (4096 BYTES)
BMC40738I       ALLOCATE PASTE BUFFER (4096 BYTES)
BMC40702I    STEP:  MERGE SPACE
BMC40738I       MERGE SPACE
BMC40301I    TABLESPACE = BMCDBSMP.BMCTS010
BMC40735I   PHASE TERMINATION:
BMC40738I       DISCONNECT FROM SPACE
BMC40738I       DEALLOCATE SPACE
BMC40345I          SPACE = BMCDBSMP.BMCTS010
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 = RDAPXR.BMCDBSMP.BMCTS010.F.T095108
BMC40738I       FREE 2 INPUT FILE BUFFER(S)
BMC40738I       DEALLOCATE INPUT IMAGE COPY FILE
BMC40302I          DSN = RDAPXR.BMCDBSMP.BMCTS010.F.T095108
BMC40738I       RECORD RESTART INFORMATION
```
Example 10: Overriding installation options

This example demonstrates the use of the OPTIONS statement to override installation options at execution time.

Example 10: Overriding installation options
Not all installation options can be overridden. See “OPTIONS” on page 113 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 syntax” on page 114.

---

### Example 10A: Overriding several installation options

In the first part of Example 10A, 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 107: Example 10A--JCL for overriding installation options**

```
//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 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 108: Example 10B--JCL for overriding the IXRECP installation option**

```
//AFREX12B JOB (PAF), 'EXAMPLE 10B',
// CLASS=Q, NOTIFY=&SYSUID,
// MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN,REGION=OM,
// 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 109 on page 430 through Figure 111 on page 435 show the following output, which is produced by running this JCL:**

- AFRSTMT
- AFRPRINT
- AFRPR001
- AFRPLAN

See Table 11 on page 326 for a description of each of these files.

**Figure 109: Example 10B--AFRSTMT for overriding the IXRECP installation option**

```
BMC40018I RECOVER PLUS FOR DB2 V11.2.00 (RECOVER AFRSTMT)
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884
7,769,718 AND 8,161,001
BMC40408I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM
BMC40093I PARM LIST: DB2 SSID = DXW
BMC40241I UTILID = EXAMPLE12B
BMC40932I UTILITY NEW/RESTART REQUESTED - SYNCPOINT RESTART WILL OCCUR IF
BMC40933I UTILID IS IN BMCUTIL
BMC40934I   RESTART = NEW/RESTART
BMC40931I MSGLEVEL = 2
BMC40930I PARM NOT CODED - CHECKPT
BMC40931I PARM NOT CODED - RDB2STAT
BMC40931I PARM NOT CODED - AFROPTS
BMC40931I PARM NOT CODED - ACAOPTS
BMC96035I ----- DEFAULT OPTIONS:
BMC40924I   PLANRECV = AFRB111T (MAIN PLAN)
BMC40408I   BINDQUALIFIER = BMCAF (QUALIFIER USED FOR
BMC40408I   DYNAMIC BIND)
BMC40931I   PUBLICPLAN = YES (GRANT PLAN TO PUBLIC)
BMC96003I   ZIIP = ENABLED (ZIIP PROCESSOR
BMC40547I   ENABLEMENT)
BMC40547I   INDEXLOG = NO (DEFAULT USE OF INDEXLOG
```
Example 10: Overriding installation options

Example 10: Overriding installation options

<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IXRECP</td>
<td>NO</td>
<td>(SET INDEX RECOVER PENDING)</td>
</tr>
<tr>
<td>BACKOUT</td>
<td>AUTO</td>
<td>(DEFAULT POINT-IN-TIME RECOVERY STRATEGY)</td>
</tr>
<tr>
<td>OPNDDB2ID</td>
<td>YES</td>
<td>(ACQUIRE RACF AUTHORITY OF DB2)</td>
</tr>
<tr>
<td>AMSCAT</td>
<td>NO</td>
<td>(RESET DB2 OBJECT STATUS)</td>
</tr>
<tr>
<td>WKUNIT</td>
<td>SYSALLDA</td>
<td>(WORK UNIT NAME)</td>
</tr>
<tr>
<td>WTOR</td>
<td>YES</td>
<td>(ISSUE WTOR IF SPACE REMAINS IN STOPP STATUS)</td>
</tr>
<tr>
<td>USEHDROBIDS</td>
<td>YES</td>
<td>(USE HEADER PAGE IDS FOR OBIDXLAT)</td>
</tr>
<tr>
<td>AUX</td>
<td>NO</td>
<td>(INCLUDE XML/LOB/ALL/AUXILIARY OBJECTS)</td>
</tr>
<tr>
<td>CHECKPT</td>
<td>PHASE</td>
<td>(CHECKPOINTING)</td>
</tr>
<tr>
<td>ERRCONT</td>
<td>10</td>
<td>(DEFAULT MAX SEVERE ERRORS)</td>
</tr>
<tr>
<td>RCLTSK</td>
<td>10</td>
<td>(MAXIMUM RECALL TASKS)</td>
</tr>
<tr>
<td>RBUFFS</td>
<td>1000</td>
<td>(BUFFER MANAGER PAGES)</td>
</tr>
<tr>
<td>HISTORY</td>
<td>YES</td>
<td>(UPDATE BMCHIST TABLE)</td>
</tr>
<tr>
<td>ACFFORTSS</td>
<td>YES</td>
<td>(USE ACF2 OR TOP SECRET SECURITY)</td>
</tr>
<tr>
<td>SMCORE</td>
<td>(OK, OK)</td>
<td>(SORT CORE VALUES)</td>
</tr>
<tr>
<td>RESINV</td>
<td>(OK)</td>
<td>(MEMORY BELOW 16MB EXCLUDED FROM USE BY SORT)</td>
</tr>
<tr>
<td>SORTDEVT</td>
<td>SORT DEFAULT</td>
<td>(DEVICE TYPE FOR LOG AND INDEX SORTS)</td>
</tr>
<tr>
<td>SORTNUM</td>
<td>SORT DEFAULT</td>
<td>(WORK DATASETS FOR LOG AND INDEX SORTS)</td>
</tr>
<tr>
<td>KSORTSHARE</td>
<td>YES</td>
<td>(KEY Sorts SHARED BY ALL TASKS)</td>
</tr>
<tr>
<td>MAXKSORT</td>
<td>10</td>
<td>(MAX CONCURRENT INDEX KEY Sorts)</td>
</tr>
<tr>
<td>MAXLSORT</td>
<td>50</td>
<td>(MAX CONCURRENT LOG Sorts)</td>
</tr>
<tr>
<td>MAXDRIVE</td>
<td>0</td>
<td>(DEFAULT MAX TAPE Drives)</td>
</tr>
<tr>
<td>MAXLOGS</td>
<td>3</td>
<td>(DEFAULT MAX CONCURRENT LOG Files)</td>
</tr>
<tr>
<td>OUTCOPY</td>
<td>ASCODED</td>
<td>(OUTCOPY CREATION AND OUTPUT ACCUM Files)</td>
</tr>
<tr>
<td>AUTOSIZE</td>
<td>YES</td>
<td>(AUTO SIZE OUTPUT COPIES)</td>
</tr>
<tr>
<td>EATTR</td>
<td>NONE</td>
<td>(EXTENDED ATTRIBUTES FOR DYNAMIC ALLOCATION OF OUTPUT COPIES)</td>
</tr>
<tr>
<td>DISKIORATE</td>
<td>100</td>
<td>(DISK READ RATE IN MB/SEC)</td>
</tr>
<tr>
<td>CPUMIPS</td>
<td>200</td>
<td>(CPU PROCESSING SPEED IN MIPS)</td>
</tr>
</tbody>
</table>

Chapter 5 Examples of RECOVER PLUS jobs 431
Example 10: Overriding installation options

432  RECOVER PLUS for DB2 Reference Manual
Example 10: Overriding installation options

Chapter 5 Examples of RECOVER PLUS jobs
Example 10: Overriding installation options

Figure 110: Example 10B--AFRPLAN for overriding the IXRECP installation option

```plaintext
BMC40001I RECOVER PLUS FOR DB2 V11.2.00 (RECOVER AFRPLAN)
BMC40018I RECOVER PLUS FOR DB2 V11.2.00 (RECOVER AFRPLAN)
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC96306I MAXLSORT REDUCED FROM 50 TO 22 DUE TO MEMORY CONSTRAINTS
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 08/21/2014 17:15:10
BMC40335I PLAN EXECUTION STARTING 08/21/2014 17:15:13
BMC40025I OUTPUT FROM COMMAND '-STA DB(BMCDBSMP) SPACE(BMCTS012)
BMC40020I DSN9022I *DXW4 DSNTDDIS 'START DATABASE' NORMAL COMPLETION
BMC40018I RECOVER PLUS FOR DB2 V11.2.00   (RECOVER  AFRPLAN)
```

Figure 110: Example 10B--AFRPLAN for overriding the IXRECP installation option

```plaintext
BMC40025I OUTPUT FROM COMMAND '-STA DB(BMCDBSMP) SPACE(BMCTS012)
BMC40020I DSN9022I *DXW4 DSNTDDIS 'START DATABASE' NORMAL COMPLETION
```

```plaintext
BMC40018I RECOVER PLUS FOR DB2 V11.2.00   (RECOVER  AFRPLAN)
```

```plaintext
BMC40700I EXECUTION PLAN SUMMARY:
BMC40701I PHASE:  MERGE
BMC40721I PHASE INITIALIZATION:
BMC40301I TABLESPACE = BMCDBSMP.BMCTS012
BMC40301I TABLESPACE = BMCDBSMP.BMCTS012
BMC40738I ALLOCATE INPUT IMAGE COPY FILE
BMC40738I ALLOCATE INPUT IMAGE COPY FILE
BMC40738I DSN = RDAPXR.BMCDBSMP.BMCTS012.F.T135427.LP
BMC40738I DSN = RDAPXR.BMCDBSMP.BMCTS012.F.T135427.LP
BMC40738I FILE OPTIMIZATION
BMC40738I FILE OPTIMIZATION
BMC40738I ALLOCATE 2 INPUT FILE BUFFER(S)
BMC40738I ALLOCATE 2 INPUT FILE BUFFER(S)
BMC40738I OPEN IMAGE COPY FILE
BMC40738I OPEN IMAGE COPY FILE
BMC40738I ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I ALLOCATE SPACEMAP BUFFER (4096 BYTES)
BMC40738I ALLOCATE SPACEMAP BUFFER (4096 BYTES)
BMC40738I ALLOCATE PASTE BUFFER (4096 BYTES)
BMC40738I ALLOCATE PASTE BUFFER (4096 BYTES)
BMC40738I SET RECOVERY PENDING (RECP) FLAG
BMC40738I SET RECOVERY PENDING (RECP) FLAG
BMC40738I CREATE SPACE
BMC40738I CREATE SPACE
BMC40345I SPACE = BMCDBSMP.BMCTS012
BMC40345I SPACE = BMCDBSMP.BMCTS012
BMC40738I ALLOCATE SPACE (READ/WRITE)
BMC40738I ALLOCATE SPACE (READ/WRITE)
BMC40738I SPACE = BMCDBSMP.BMCTS012
BMC40738I SPACE = BMCDBSMP.BMCTS012
BMC40738I CONNECT TO SPACE
BMC40738I CONNECT TO SPACE
BMC40738I ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I FILE OPTIMIZATION
BMC40738I FILE OPTIMIZATION
BMC40738I ALLOCATE 2 SPACE OUTPUT BUFFER(S)
BMC40738I ALLOCATE 2 SPACE OUTPUT BUFFER(S)
BMC40738I ALLOCATE SPACEMAP BUFFER (4096 BYTES)
BMC40738I ALLOCATE SPACEMAP BUFFER (4096 BYTES)
BMC40738I ALLOCATE PASTE BUFFER (4096 BYTES)
BMC40738I ALLOCATE PASTE BUFFER (4096 BYTES)
BMC40702I STEP:  MERGE SPACE
BMC40702I STEP:  MERGE SPACE
BMC40702I MERGE SPACE
BMC40702I MERGE SPACE
BMC40738I TABLESPACE = BMCDBSMP.BMCTS012
BMC40738I TABLESPACE = BMCDBSMP.BMCTS012
BMC40735I PHASE TERMINATION:
BMC40735I PHASE TERMINATION:
BMC40738I DISCONNECT FROM SPACE
BMC40738I DISCONNECT FROM SPACE
BMC40345I SPACE = BMCDBSMP.BMCTS012
BMC40345I SPACE = BMCDBSMP.BMCTS012
BMC40738I FREE PAGE CHECK BUFFER
BMC40738I FREE PAGE CHECK BUFFER
BMC40738I FREE SPACEMAP BUFFER
BMC40738I FREE SPACEMAP BUFFER
BMC40738I FREE PASTE BUFFER
BMC40738I FREE PASTE BUFFER
BMC40738I FREE 2 OUTPUT SPACE BUFFER(S)
BMC40738I FREE 2 OUTPUT SPACE BUFFER(S)
BMC40738I FREE PAGE CHECK BUFFER
BMC40738I FREE PAGE CHECK BUFFER
BMC40738I FREE SPIECMAP BUFFER
BMC40738I FREE SPACEMAP BUFFER
BMC40738I CLOSE IMAGE COPY FILE
BMC40302I DSN = RDAPXR.BMCDBSMP.BMCTS012.F.T135427.LP
```
Figure 111: Example 10B--AFRPRINT and AFRPR001 for overriding the IXRECP installation option

BMC40016I MERGE PHASE STARTING 08/21/2014 17:15:13
BMC40301I TABLESPACE = BMCDBSMP.BMCTS012
BMC40350I DATA SET RDAPXR.BMCDBSMP.BMCTS012.F.T135427.LP WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00002 ELAPSED = 00:00:00
BMC40060I PENDING FLAG RECP SET FOR
BMC40345I SPACE = BMCDBSMP.BMCTS012
BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.BMCDBSMP.BMCTS012.I0001.A001
BMC40356I AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DSNDXW.DSNDBC.BMCDBSMP.BMCTS012.I0001.A001
BMC40350I DATA SET DSNDXW.DSNDBD.BMCDBSMP.BMCTS012.I0001.A001 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00004 ELAPSED = 00:00:00
BMC96304I THIS PHASE WILL RUN IN EXECUTION SUBTASK 1
BMC40593I INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00
BMC40593I SPIN COMPLETE, ELAPSED TIME = 00:00:00
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE BMCDBSMP.BMCTS012
BMC40302I DSN = DSNDXW.DSNDBD.BMCDBSMP.BMCTS012.I0001.A001
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE BMCDBSMP.BMCTS012
BMC40302I DSN = DSNDXW.DSNDBD.BMCDBSMP.BMCTS012.I0001.A001
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE BMCDBSMP.BMCTS012
BMC40302I DSN = DSNDXW.DSNDBD.BMCDBSMP.BMCTS012.I0001.A001
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE BMCDBSMP.BMCTS012
BMC40302I DSN = DSNDXW.DSNDBD.BMCDBSMP.BMCTS012.I0001.A001
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE BMCDBSMP.BMCTS012
BMC40302I DSN = DSNDXW.DSNDBD.BMCDBSMP.BMCTS012.I0001.A001
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE BMCDBSMP.BMCTS012
BMC40302I DSN = DSNDXW.DSNDBD.BMCDBSMP.BMCTS012.I0001.A001
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE BMCDBSMP.BMCTS012
BMC40302I DSN = DSNDXW.DSNDBD.BMCDBSMP.BMCTS012.I0001.A001
BMC40902I TOTAL WAITS = 26, TOTAL WAIT TIME = 00:00:00
BMC40981I FULL IC RDAPXR.BMCDBSMP.BMCTS012.F.T135427.LP READ WAITS = 4, WAIT TIME = 00:01:00
BMC40981I FULL IC RDAPXR.BMCDBSMP.BMCTS012.F.T135427.LP WRITE WAITS = 22, WRITE TIME = 00:00:00
BMC40981I NUMBER OF PAGES READ FROM FULL COPY FOR THIS OBJECT = 5003, USED
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 the following figure creates the change accumulation file.

Figure 112: Example 11A--JCL using R+/CHANGE ACCUM to accumulate log records

Example 11B: Using the change accumulation file

The JCL in the following figure recovers the table space using the change accumulation file created in Example 11A.
Example 12: Using the MAINT parameter

Example 12 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.

Example 13: Using the LOGSCAN command

Example 13 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 SYSSCAN DD statement included in this example to designate output for the report that the LOGSCAN command generates.

**Figure 115: Example 13--JCL using the LOGSCAN command**

```jcl
//AFREX15 JOB (PAFR), 'EXAMPLE 13',
//       CLASS=Q, NOTIFY=&SYSUID,
//       MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN, REGION=0M,
//       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 is shown in the following figure. AFRPRINT and AFRPR001 are shown in Figure 117 on page 439. See Table 11 on page 326 for a description of each of these files.

**Figure 116: Example 13--SYSSCAN for the LOGSCAN command**

```
BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40860I LOG SORT STATISTICS (LOGSCAN)
BMC40861I 08/21/2014 17:45:41

BMC40512I LOG FILE RESOURCES:
BMC40302I DSN = DSNDXW.DXW1.LOGCOPY1.DS02.DATA
BMC40302I DSN = DSNDXW.DXW3.LOGCOPY1.DS02.DATA
BMC40302I DSN = DSNDXW.DXW4.LOGCOPY1.DS03.DATA
BMC40302I DSN = DSNDXW.DXW2.LOGCOPY1.DS06.DATA

BMC40566I USING LOG DSNDXW.DXW1.LOGCOPY1.DS02.DATA FOR PROCESSING
BMC40555I FROMRBA = X'000000000222C08A8000'  TORBA =
X'000000000222CB86FFFF'
BMC40566I USING LOG DSNDXW.DXW3.LOGCOPY1.DS02.DATA FOR PROCESSING
BMC40555I FROMRBA = X'00000000018770CA3000'  TORBA =
X'0000000001877BC6AFFF'
BMC40566I USING LOG DSNDXW.DXW4.LOGCOPY1.DS03.DATA FOR PROCESSING
BMC40555I FROMRBA = X'00000000015BA0F0D000'  TORBA =
X'00000000015BABED4FFF'
BMC40566I USING LOG DSNDXW.DXW2.LOGCOPY1.DS06.DATA FOR PROCESSING
BMC40555I FROMRBA = X'0000000001C91E5FB000'  TORBA =
X'0000000001C9DFBBAFFF'

BMC40189I THE 0 TAPE AND 4 DISK LOG FILES WILL BE READ IN 2 LOG READING
STEPS BECAUSE MAXDRIVES = 0 AND MAXLOGS = 3

BMC40850I OBJECT TYPE PART #RECS #BYTES
BMC40551I BMCDBSMP.BMCTS014 TS 000 0 0
BMC40551I BMCDBSMP.BMCIX014 IX 000 0 0

BMC40852I TOTAL 0 0
BMC40853I NUMREC EST 0 AVERAGE SIZE 0
```

Example 13: Using the LOGSCAN command

438 RECOVER PLUS for DB2 Reference Manual
**Figure 117: Example 13--AFRPRINT and AFRPR001**

```
BMC40016I LOGSCAN PHASE STARTING 08/21/2014 17:45:44
BMC40725I PROCESSING LOG RANGE:
BMC40302I   DSN = DSNDXW.DXW2.LOGCOPY1.DS06.DATA
BMC40555I   FROMRBA = X'0000000001C91E5FB000'  TORBA =
X'0000000001C9DFBBAFFF'
BMC40724I   DATA SHARING MEMBERID IS 4
BMC40725I PROCESSING LOG RANGE:
BMC40302I   DSN = DSNDXW.DXW4.LOGCOPY1.DS03.DATA
BMC40555I   FROMRBA = X'00000000015BA0F0D000'  TORBA =
X'00000000015BABED4FFF'
BMC40724I   DATA SHARING MEMBERID IS 3
BMC40725I PROCESSING LOG RANGE:
BMC40302I   DSN = DSNDXW.DXW1.LOGCOPY1.DS02.DATA
BMC40555I   FROMRBA = X'000000000222C08A8000'  TORBA =
X'000000000222C08B6FFFF'
BMC40724I   DATA SHARING MEMBERID IS 1
BMC40193I   3 LOG DATASET BUFFERS OF 348160 BYTES EACH WERE ALLOCATED
BMC40302I   DSN = DSNDXW.DXW4.LOGCOPY1.DS03.DATA
BMC40173I   TIME SPENT IN DATASET ALLOCATION = 00:00:00
BMC40302I   DSN = DSNDXW.DXW1.LOGCOPY1.DS02.DATA
BMC40173I   TIME SPENT IN DATASET ALLOCATION = 00:00:00
BMC40302I   DSN = DSNDXW.DXW2.LOGCOPY1.DS06.DATA
BMC40352I  DDNAME SYS00012 (DSN DSNDXW.DXW1.LOGCOPY1.DS02.DATA) WAS
SUCCESSFULLY UNALLOCATED
BMC40352I  DDNAME SYS00010 (DSN DSNDXW.DXW4.LOGCOPY1.DS03.DATA) WAS
SUCCESSFULLY UNALLOCATED
BMC40352I  DDNAME SYS00011 (DSN DSNDXW.DXW2.LOGCOPY1.DS06.DATA) WAS
SUCCESSFULLY UNALLOCATED

BMC40117I ********** PERFORMANCE STATISTICS FROM LOG READ STEP
************
BMC40120I   3 FILES READ   21542 I/O WAITS   I/O WAIT TIME =
00:00:30
BMC40175I   4 WAITS FOR THREAD   WAIT TIME = 00:00:00
BMC40172I   0 LOG RECORDS AND 0 BYTES OF LOG DATA
BMC40173I   TIME SPENT IN DATASET ALLOCATION = 00:00:00
BMC40174I   21237 I/O WAITS   I/O WAIT TIME = 00:00:30
BMC40172I   0 LOG RECORDS AND 0 BYTES OF LOG DATA
BMC40173I   TIME SPENT IN DATASET ALLOCATION = 00:00:00
BMC40174I   13 I/O WAITS   I/O WAIT TIME = 00:00:00
BMC40172I   0 LOG RECORDS AND 0 BYTES OF LOG DATA
```

Example 13: Using the LOGSCAN command

Chapter 5  Examples of RECOVER PLUS jobs  439
Example 14: Simulating recovery

Example 14 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 118: Example 14--JCL using the SIMULATE option

```
//AFREX1B JOB (PAFR), 'EXAMPLE 14',
//      CLASS=Q, NOTIFY=&SYSUID,
//      MSGCLASS=X
/*ROUTE XEQ BMCPLX1
/*JOBPARM SYSAFF=DB2A
/**
//RECOVER EXEC PGM=AFRMAIN, REGION=OM,
//      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 *
```
Example 15: Using MAXKSORT for parallel index rebuilds

Example 15 illustrates the use of the MAXKSORT option to invoke parallel index sorts and rebuilds to rebuild a partitioned index and multiple nonpartitioned indexes.

Figure 119: Example 15--JCL using the MAXKSORT option for parallel index rebuilds

```
//AFREX16  JOB (PAFR),'EXAMPLE 15',
  //         CLASS=Q,NOTIFY=&SYSUID,
  //         MSGCLASS=X,TIME=(0,5)
/*ROUTE XEQ BMCPLX1
/*JOBPARM SYSAFF=DB2A
//*
//RECOVER EXEC PGM=AFRMAIN,REGION=0M,
//         PARM='DEBA,EXAMPLE15,NEW/RESTART,MSGLEVEL(0)'
//STEPLIB  DD DISP=SHR,DSN=product.libraries
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//         DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSIN     DD *
OPTIONS MAXKSORT 12
BMC40101I   REBUILD INDEX (ALL) TABLESPACE RMDSUPDB.TS67P32K
/*
/*
```

Figure 121 on page 442 through Figure 123 on page 447 show the following output, which is produced by running this JCL:

- AFRERR
- AFRSUMRY
- AFRSTMT
- AFRPRINT
- AFRPR001 (AFRPR002 through AFRPR009 are produced by not shown.

See Table 11 on page 326 for a description of each of these files.

Figure 120: Example 15--AFRSUMRY for parallel index rebuilds with MAXKSORT

```
BMC40018I RECOVER PLUS FOR DB2 V11.2.00
```
Example 15: Using MAXKSORT for parallel index rebuilds

Figure 121: Example 15--AFERR for parallel index rebuilds with MAXKSORT

Figure 122: Example 15--AFRSTMT for parallel index rebuilds with MAXKSORT
Example 15: Using MAXKSORT for parallel index rebuilds
Example 15: Using MAXKSORT for parallel index rebuilds

DIRECTIVE
BMC96221I AUTOSIZE = YES (AUTO SIZE OUTPUT COPIES AND OUTPUT ACCUM FILES)
BMC96033I EATTR = NONE (EXTENDED ATTRIBUTES FOR DYNAMIC ALLOCATION OF OUTPUT COPIES)
BMC96035I ----- DEFAULT OPTIONS: ESTIMATION
BMC96251I DISKIORATE = 100 (DISK READ RATE IN MB/SEC)
BMC96252I CPUMIPS = 200 (CPU PROCESSING SPEED IN MIPS)
BMC96035I ----- DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE
BMC96113I COPIES = (FC,LP,LB) (DEFAULT SEQUENCE FOR IMAGE COPIES)
BMC96114I ACCUMS = (LP,LB) (DEFAULT SEQUENCE FOR CA FILES)
BMC96112I LOGS = (ACT1,ACT2) (DEFAULT SEQUENCE FOR LOG FILES)
BMC96112I ----- DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE
BMC96113I COPIES = (RP,RB,FC) (DEFAULT SEQUENCE FOR IMAGE COPIES)
BMC96114I ACCUMS = (RP,RB) (DEFAULT SEQUENCE FOR CA FILES)
BMC96112I LOGS = (ACT1,ACT2) (DEFAULT SEQUENCE FOR LOG FILES)
BMC96112I ----- TAPE DEVICES
BMC40967I TAPE DEVICE01 = 3590-1
BMC40967I TAPE DEVICE02 = 3490
BMC40967I TAPE DEVICE03 = 3480X
BMC40967I TAPE DEVICE04 = ATLYTS
BMC40967I TAPE DEVICE05 = AT3592
BMC40967I TAPE DEVICE06 = CART
BMC40967I TAPE DEVICE07 = CARTVTS
BMC40967I TAPE DEVICE08 = CRTBNS
BMC40967I TAPE DEVICE09 = CRT9NS
BMC40967I TAPE DEVICE10 = EXT3592
BMC40967I TAPE DEVICE11 = IBMATL
BMC40967I TAPE DEVICE12 = IBMVTS
BMC40967I TAPE DEVICE13 = PHXATL
BMC40967I TAPE DEVICE14 = SA3590
BMC40967I TAPE DEVICE15 = TAPEABLF
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

BMC40020I EBCDIC CCSID = 37

Example 15: Using MAXKSORT for parallel index rebuilds

DIRECTIVE
BMC96221I AUTOSIZE = YES (AUTO SIZE OUTPUT COPIES AND OUTPUT ACCUM FILES)
BMC96033I EATTR = NONE (EXTENDED ATTRIBUTES FOR DYNAMIC ALLOCATION OF OUTPUT COPIES)
BMC96035I ----- DEFAULT OPTIONS: ESTIMATION
BMC96251I DISKIORATE = 100 (DISK READ RATE IN MB/SEC)
BMC96252I CPUMIPS = 200 (CPU PROCESSING SPEED IN MIPS)
BMC96035I ----- DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE
BMC96113I COPIES = (FC,LP,LB) (DEFAULT SEQUENCE FOR IMAGE COPIES)
BMC96114I ACCUMS = (LP,LB) (DEFAULT SEQUENCE FOR CA FILES)
BMC96112I LOGS = (ACT1,ACT2) (DEFAULT SEQUENCE FOR LOG FILES)
BMC96112I ----- DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE
BMC96113I COPIES = (RP,RB,FC) (DEFAULT SEQUENCE FOR IMAGE COPIES)
BMC96114I ACCUMS = (RP,RB) (DEFAULT SEQUENCE FOR CA FILES)
BMC96112I LOGS = (ACT1,ACT2) (DEFAULT SEQUENCE FOR LOG FILES)
BMC96112I ----- TAPE DEVICES
BMC40967I TAPE DEVICE01 = 3590-1
BMC40967I TAPE DEVICE02 = 3490
BMC40967I TAPE DEVICE03 = 3480X
BMC40967I TAPE DEVICE04 = ATLYTS
BMC40967I TAPE DEVICE05 = AT3592
BMC40967I TAPE DEVICE06 = CART
BMC40967I TAPE DEVICE07 = CARTVTS
BMC40967I TAPE DEVICE08 = CRTBNS
BMC40967I TAPE DEVICE09 = CRT9NS
BMC40967I TAPE DEVICE10 = EXT3592
BMC40967I TAPE DEVICE11 = IBMATL
BMC40967I TAPE DEVICE12 = IBMVTS
BMC40967I TAPE DEVICE13 = PHXATL
BMC40967I TAPE DEVICE14 = SA3590
BMC40967I TAPE DEVICE15 = TAPEABLF
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

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40876I MAINT: NO RECOVER PLUS PTFS APPLIED

BMC96010I SOLUTION COMMON CODE V11.2.01
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFS APPLIED

BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0199 BPJ0209 BPJ0250 BPJ0263 BPJ0269
BPJ0279
BMC40876I MAINT: BPJ0288 BPJ0308 BPJ0343 BPJ0357 BPJ0361 BPJ0374
BPJ0590
BMC40876I MAINT: BPJ0404 BPJ0473 BPJ0503 BPJ0521

BMC40335I PACKAGE CHECKING AND BINDING STARTING 08/21/2014 18:05:06
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
BMC40020I EBCDIC CCSID = 37
Example 15: Using MAXKSORT for parallel index rebuilds

BMC96014I  DB2 VERSION = 910  SITE TYPE = LOCAL  MODE = NFM
BMC40954I  AVAILABLE MEMORY BELOW THE LINE = 9044K, TOTAL AVAILABLE MEMORY
           AT LEAST = 1414552K
BMC40037I  UTILID EXAMPLE16 WAS NOT FOUND IN THE BMCUTIL TABLE - UTILITY
           WILL RUN AS IF 'NEW' WERE SPECIFIED
BMC40611I  INPUT STATEMENTS:
BMC40101I   OPTION MAXKSORT 12
BMC40101I   REBUILD INDEX (ALL) TABLESPACE RMDSUPDB.TS67P32K
BMC40957I  MAXKSORT VALUE OVERRIDDEN, NEW VALUE = 12
BMC96004I  ZIIP ENABLED (0) USING SUBSYSTEM XBMA
BMC40937I  z/OS RELEASE = 1.13.00, PID = HBB7780, SMS RELEASE = 1.13.0
BMC40157I  AVAILABLE REGION BELOW 16M = 9032K, AVAILABLE REGION ABOVE 16M =
           1404328K, NUMBER OF CPUS = 5
BMC40020I  BYT:0, 0, 0; BLK:0, 0, 0; TOT:1656364, 10037136, 10037136; iter:
           1; RC:4
BMC40335I  ANALYZE STARTING 08/21/2014 18:05:06
BMC40335I  CATALOG LOOKUP STARTING 08/21/2014 18:05:06
BMC40336I  PROC IX STMFT 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 SYSIX 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  COL 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 SYSKY 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 SYSFL 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
BMC40068I  THE KEYS FOR INDEX RDAPXR.IX67P324 WILL GO DIRECTLY TO THE SORT
           BECAUSE THERE IS NO SYSUT1 DD STATEMENT IN THE JCL
BMC40068I  THE KEYS FOR INDEX RDAPXR.IX67P323 WILL GO DIRECTLY TO THE SORT
           BECAUSE THERE IS NO SYSUT1 DD STATEMENT IN THE JCL
BMC40068I  THE KEYS FOR INDEX RDAPXR.IX67P322 WILL GO DIRECTLY TO THE SORT
           BECAUSE THERE IS NO SYSUT1 DD STATEMENT IN THE JCL
BMC40068I  THE KEYS FOR INDEX RDAPXR.IX67P32 WILL GO DIRECTLY TO THE SORT
           BECAUSE THERE IS NO SYSUT1 DD STATEMENT IN THE JCL
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
Example 15: Using MAXKSORT for parallel index rebuilds

446  RECOVER PLUS for DB2 Reference Manual
BMC40025I OUTPUT FROM COMMAND '-STA DB(RMDSUPDB) SPACE(IX67P32) ACCESS(RW)' FOLLOWS:
BMC40020I DSN9022I *DEDL DSNTDDIS 'START DATABASE' NORMAL COMPLETION

Figure 123: Example 15--AFRPRINT for parallel index rebuilds with MAXKSORT

BMC40018I RECOVER PLUS FOR DB2 V11.2.00 (RECOVER AFRPRINT)
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884
7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC96400I UNLOAD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR001
BMC40301I TABLESPACE = RMDSUPDB.TS67P32K
BMC40305I DSNUM = 1
BMC96400I UNLOAD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR002
BMC40301I TABLESPACE = RMDSUPDB.TS67P32K
BMC40305I DSNUM = 2
BMC96400I UNLOAD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR003
BMC40301I TABLESPACE = RMDSUPDB.TS67P32K
BMC40305I DSNUM = 3
BMC96400I UNLOAD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR004
BMC40301I TABLESPACE = RMDSUPDB.TS67P32K
BMC40305I DSNUM = 4
BMC96400I UNLOAD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR005
BMC40301I TABLESPACE = RMDSUPDB.TS67P32K
BMC40305I DSNUM = 5
BMC96400I UNLOAD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR006
BMC40301I TABLESPACE = RMDSUPDB.TS67P32K
BMC40305I DSNUM = 6
BMC96400I BUILD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR001
BMC40303I INDEX = RDAPXR.IX67P32
BMC40305I DSNUM = 1
BMC96400I BUILD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR002
BMC40303I INDEX = RDAPXR.IX67P32
BMC40305I DSNUM = 2
BMC96400I BUILD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR003
BMC40303I INDEX = RDAPXR.IX67P32
BMC40305I DSNUM = 3
BMC96400I BUILD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR004
BMC40303I INDEX = RDAPXR.IX67P32
BMC40305I DSNUM = 4
BMC96400I BUILD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR005
BMC40303I INDEX = RDAPXR.IX67P32
BMC40305I DSNUM = 5
BMC96400I BUILD PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR006
BMC40303I INDEX = RDAPXR.IX67P32
BMC40305I DSNUM = 6

Example 15: Using MAXKSORT for parallel index rebuilds
In addition to AFRPR001 in the following figure, this job also produces AFRPR002 through AFRPR009.

Figure 124: Example 15--AFRPR001 for parallel index rebuilds with MAXKSORT

Example 15: Using MAXKSORT for parallel index rebuilds
Example 15: Using MAXKSORT for parallel index rebuilds

Chapter 5  Examples of RECOVER PLUS jobs  449
Example 16: Using MAXKSORT and recovering a table space

Example 16 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 125: Example 16--JCL for table space recovery and index recovery using the MAXKSORT option**

```plaintext
//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.

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

The following figure is an example of RECOVER PLUS JCL that includes the options for a timestamp recovery.

**Figure 126: Example 17--JCL using RECOVERYPOINT for timestamp recovery**

```sql
//AFREX18 JOB (PAFR), 'EXAMPLE 17',
//       CLASS=Q, NOTIFY=&SYSUID,
//       MSGCLASS=X
//*
//RECOVER EXEC PGM=AFRMAIN, REGION=OM,
//       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 WRPDB01.WRPTS01
RECOVER INDEX(ALL) TABLESPACE WRPDB01.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 127: Example 18--JCL using default values**

```plaintext
//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=product.libraries
//  DD DISP=SHR, DSN=DB2.DSNEXIT
//  DD DISP=SHR, DSN=DB2.DSNLOAD
//SYSIN DD *
OPTIONS MAXLSORT 50
RECOVER TABLESPACE G4GLMQDB.V9LMQTS1
/*
//
```

**Figure 128 on page 452 through Figure 135 on page 463** show the following output, which is produced by running this JCL:

- AFRSTMT
- AFRSUMRY
- AFRPRINT
- AFRPR001
- AFRPR002
- AFRPR003
- AFROSUM
- AFPLAN

See “DD statements common to all RECOVER PLUS executions” on page 326 for a description of each of these files.

The AFRSTMT output shown in the following figure 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 and MAXKSORT are also highlighted in this output.

**Figure 128: Example 18--AFRSTMT showing the MAXLSORT value**
Example 18: Using the MAXLSORT default value

**Chapter 5  Examples of RECOVER PLUS jobs**  453

| BMC40934I | MSGLEVEL = 2 |
| BMC40930I | PARM NOT CODED - CHECKPT |
| BMC40930I | PARM NOT CODED - RDB2STAT |
| BMC40930I | PARM NOT CODED - AFROPTS |
| BMC40930I | PARM NOT CODED - ACAOPTS |
| BMC96035I | ------ DEFAULT OPTIONS: |
| BMC40924I | PLANRECV = AFRB111T (MAIN PLAN) |
| BMC40408I | BINDQUALIFIER = BMCAF |
| BMC40241I | PUBLICPLAN = YES (GRANT PLAN TO PUBLIC) |
| BMC96003I | ZIIP = ENABLED (ZIIP PROCESSOR) |
| BMC40941I | IXRECP = NO (SET INDEX RECOVER PENDING) |
| BMC96233I | BACKOUT = AUTO (DEFAULT POINT-IN-TIME RECOVERY STRATEGY) |
| BMC40944I | OPNDB2ID = YES (ACQUIRE RACF AUTHORITY) |
| BMC40927I | RDB2STAT = YES (RESET DB2 OBJECT STATUS) |
| BMC40946I | AMSCAT = NO (USE 'CATALOG' IN IDCAMS INPUT) |
| BMC40938I | WKUNIT = SYSALLDA (WORK UNIT NAME) |
| BMC40878I | WTOR = YES (ISSUE WTOR IF SPACE REMAINS IN STOUP STATUS) |
| BMC96034I | USEHDROBIDS = YES (USE HEADER PAGE IDS FOR OBIDXLAT) |
| BMC96235I | AUX = NO (INCLUDE XML/LOB/ALL/ HISTORY AUXILIARY OBJECTS) |
| BMC40926I | CHECKPT = PHASE (CHECKPOINTING) |
| BMC40945I | CHECKINT = 0 (CHECKPOINT INTERVAL MINUTES) |
| BMC40537I | ERRCONT = 10 (DEFAULT MAX SEVERE ERRORS) |
| BMC40956I | RCLTSK = 10 (MAXIMUM RECALL TASKS) |
| BMC40925I | TBUFFS = 1000 (BUFFER MANAGER PAGES) |
| BMC96245I | HISTORY = YES (UPDATE BMCHIST TABLE) |
| BMC96246I | ACFORTSS = YES (USE ACF2 OR TOP SECRET SECURITY) |
| BMC96035I | ------ DEFAULT OPTIONS: SORT |
| BMC40928I | SMCORE = (OK, OK) (SORT CORE VALUES) |
| BMC40893I | RESINV = (OK) (MEMORY BELOW 16MB EXCLUDED FROM USE BY SORT) |
| BMC40962I | SORTDEVT = SORT DEFAULT (DEVICE TYPE FOR LOG AND INDEX SORTS) |
| BMC40963I | SORTNUM = SORT DEFAULT (WORK DATASETS FOR LOG AND INDEX SORTS) |
| BMC96230I | KSORTSHARE = YES (KEY SORTS SHARED BY ALL TASKS) |
| BMC40942I | MAXKSORT = 10 (DEFAULT) (MAX CONCURRENT INDEX KEY SORTS) |
| BMC96192I | MAXLSORT = 50 (MAX CONCURRENT LOG SORTS) |
| BMC40891I | MAXDRIVE = 0 (DEFAULT MAX TAPE DRIVES) |
| BMC40892I | MAXLOGS = 3 (DEFAULT MAX CONCURRENT LOG FILES) |
| BMC96035I | ------ DEFAULT OPTIONS: OUTPUT COPIES |
| BMC40964I | OUTCOPY = ASCODED (OUTCOPY CREATION DIRECTIVE) |
| BMC96221I | AUTOSIZE = YES (AUTO SIZE OUTPUT COPIES AND OUTPUT ACCUM FILES) |
| BMC96033I | EATTR = NONE (EXTENDED ATTRIBUTES FOR DYNAMIC ALLOCATION OF OUTPUT COPIES) |
| BMC96035I | ------ DEFAULT OPTIONS: ESTIMATION |
Example 18: Using the MAXLSORT default value

BMC96251I DISK I/O RATE = 100 (DISK READ RATE IN MB/SEC)
BMC96252I CPU MIPs = 200 (CPU PROCESSING SPEED IN MIPS)
BMC96035I ----- DEFAULT RESOURCE SELECTION SEQUENCE FOR LOCAL SITE
BMC96113I COPIES = (FC, LP, LB) (DEFAULT SEQUENCE FOR IMAGE COPIES)
BMC96114I ACCUMS = (LP, LB) (DEFAULT SEQUENCE FOR CA FILES)
BMC96112I LOGS = (ACT1, ACT2) (DEFAULT SEQUENCE FOR LOG FILES)
BMC96112I ARC1, ARC2)
BMC96035I ----- DEFAULT RESOURCE SELECTION SEQUENCE FOR REMOTE SITE
BMC96113I COPIES = (RP, RB, FC) (DEFAULT SEQUENCE FOR IMAGE COPIES)
BMC96114I ACCUMS = (RP, RB) (DEFAULT SEQUENCE FOR CA FILES)
BMC96112I LOGS = (ACT1, ACT2) (DEFAULT SEQUENCE FOR LOG FILES)
BMC96112I ARC1, ARC2)
BMC96035I ----- TAPE DEVICES
BMC40967I TAPE DEVICE01 = 3590-1
BMC40967I TAPE DEVICE02 = 3490
BMC40967I TAPE DEVICE03 = 3480X
BMC40967I TAPE DEVICE04 = ATLVT5
BMC40967I TAPE DEVICE05 = ATL3592
BMC40967I TAPE DEVICE06 = CART
BMC40967I TAPE DEVICE07 = CARTVT5
BMC40967I TAPE DEVICE08 = CRTBNS
BMC40967I TAPE DEVICE09 = CRT9NS
BMC40967I TAPE DEVICE10 = EXT3592
BMC40967I TAPE DEVICE11 = IBMATL
BMC40967I TAPE DEVICE12 = IBMVT5
BMC40967I TAPE DEVICE13 = PHXATL
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
BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40876I MAINT: NO RECOVER PLUS PTFS APPLIED
BMC96010I SOLUTION COMMON CODE V11.2.01
BMC40876I MAINT: NO SOLUTION COMMON CODE PTFS APPLIED
BMC96010I BMCSORT ENGINE V2.3.01
BMC40876I MAINT: BPJ0195 BPJ0198 BPJ0209 BPJ0250 BPJ0263 BPJ0269 BPJ0279
BMC40876I MAINT: BPJ0288 BPJ0308 BPJ0343 BPJ0357 BPJ0361 BPJ0374 BPJ0390
BMC40876I MAINT: BPJ0404 BPJ0473 BPJ0503 BPJ0521
BMC40335I PACKAGE CHECKING AND BINDING STARTING 08/22/2014 17:16:43
BMC40356I 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
BMC40020I EBCDIC CCSID = 37
BMC96014I DB2 VERSION = 910 SITE TYPE = LOCAL MODE = NFM
BMC40954I AVAILABLE MEMORY BELOW THE LINE = 9044K, TOTAL AVAILABLE MEMORY (AT LEAST) = 1414556K
BMC40611I INPUT STATEMENTS:
BMC40101I OPTIONS MAXLSORT 50
Example 18: Using the MAXLSORT default value

BMC40101I RECOVER TABLESPACE G4GLMQDB.V9LMQTS1
BMC40101I
BMC40957I MAXLSORT VALUE OVERRIDDEN. NEW VALUE = 50
BMC96004I ZIIP ENABLED (0) USING SUBSYSTEM XBMA
BMC40937I z/OS RELEASE = 1.13.00, PID = HBB7780, SMS RELEASE = 1.13.0
BMC40157I AVAILABLE REGION BELOW 16M = 9032K, AVAILABLE REGION ABOVE 16M = 1404328K, NUMBER OF CPUS = 5
BMC40020I BYT:0, 0, 0; BLK:0, 0, 0; TOT:1912748, 9558608, 9558608;iter: 1; RC:4
BMC40335I ANALYZE STARTING 08/22/2014 17:16:43
BMC40335I CATALOG LOOKUP STARTING 08/22/2014 17:16:43
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40336I SCAN SYSTB FINISHED.
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40336I SCAN SYSIX FINISHED.
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40336I TS LOOKUP FINISHED.
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40336I DB LOOKUP FINISHED.
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40336I TB LOOKUP FINISHED.
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40336I SG LOOKUP FINISHED.
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40705I VOLUMES IN STOGROUP G4GLMQSG WILL BE USED IN THE FOLLOWING ORDER:
BMC40707I (<SMS VOLUME>)
BMC40336I SCAN SYSVOL FINISHED.
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40336I CHK DS FINISHED.
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40336I HDR PG RD FINISHED.
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40336I SYSCOPY LOOKUP FINISHED.
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40336I CATALOG LOOKUP FINISHED.
BMC40287I CPU TIME = 00:00:00, UTILITY CPU TIME = 00:00:00
BMC40288I
BMC40336I SYNCHRONIZATION STARTING 08/22/2014 17:16:43
Example 18: Using the MAXLSORT default value

Example 18: Using the MAXLSORT default value

Figure 129: Example 18--AFRSUMRY

Figure 129: Example 18--AFRSUMRY

Figure 129: Example 18--AFRSUMRY

Figure 129: Example 18--AFRSUMRY

Figure 129: Example 18--AFRSUMRY

Figure 129: Example 18--AFRSUMRY

Figure 129: Example 18--AFRSUMRY

Figure 129: Example 18--AFRSUMRY

Figure 129: Example 18--AFRSUMRY

Figure 129: Example 18--AFRSUMRY

Figure 129: Example 18--AFRSUMRY

Figure 129: Example 18--AFRSUMRY
The BMC96400I messages highlighted in AFRPRINT in the following figure point to the AFRPRxxx output files. The BMC96304I messages highlighted in the AFRPR001, AFRPR002, and AFRPR003 in Figure 131 on page 457 through Figure 133 on page 460 indicate the use of parallel merges.

**Figure 130: Example 18--AFRPRINT**

```plaintext
BMC96400I MERGE PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR001
BMC40301I TABLSPACE = G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 1

BMC96400I MERGE PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR002
BMC40301I TABLSPACE = G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2

BMC96400I MERGE PHASE MESSAGES FOR THE FOLLOWING OBJECT ARE IN FILE AFRPR003
BMC40301I TABLSPACE = G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 3
```

**Figure 131: Example 18--AFRPR001**

```plaintext
BMC40016I MERGE PHASE STARTING 08/22/2014 17:16:49
BMC40301I TABLSPACE = G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 1
BMC40350I DATA SET AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P01 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00004 ELAPSED = 00:00:00
BMC40345I SPACE = G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 1
BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DEDLCAT.DSNDBC.G4GLMQDB.V9LMQTS1.I0001.A001
BMC40356I AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DEDLCAT.DSNDBC.G4GLMQDB.V9LMQTS1.I0001.A001
BMC40350I DATA SET DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A001 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00004 ELAPSED = 00:00:00
```

Chapter 5  Examples of RECOVER PLUS jobs  457
SUCCESSFULLY ALLOCATED TO DDNAME SYS00010 ELAPSED = 00:00:00

BMC96304I THIS PHASE WILL RUN IN EXECUTION SUBTASK 1
BMC40593I INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00
BMC40593I SPIN COMPLETE, ELAPSED TIME = 00:00:00
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I        DSNUM = 1
BMC40302I        DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A001
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I        DSNUM = 1
BMC40302I        DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A001
BMC40597I PROGRESS--PAGE NUMBER 100000, ELAPSED TIME = 00:00:05
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I        DSNUM = 1
BMC40302I        DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A001
BMC40597I PROGRESS--PAGE NUMBER 200000, ELAPSED TIME = 00:00:12
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I        DSNUM = 1
BMC40302I        DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A001
BMC40597I PROGRESS--PAGE NUMBER 300000, ELAPSED TIME = 00:00:17
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I        DSNUM = 1
BMC40302I        DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A001
BMC40597I PROGRESS--PAGE NUMBER 400000, ELAPSED TIME = 00:00:22
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I        DSNUM = 1
BMC40302I        DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A001
BMC40902I TOTAL WAITS = 2354, TOTAL WAIT TIME = 00:00:21
BMC40981I FULL IC AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P01 READ WAITS = 183, WAIT TIME = 00:00:00
BMC40905I SPACE WRITE WAITS = 2171, WAIT TIME = 00:00:20
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 = 00:00:24
BMC40352I DDNAME SYS00010 (DSN DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A001) WAS SUCCESSFULLY UNALLOCATED
BMC40352I DDNAME SYS00004 (DSN AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P01) WAS SUCCESSFULLY UNALLOCATED
BMC40060I PENDING FLAG RECP RESET FOR
BMC40345I SPACE = G4GLMQDB.V9LMQTS1
BMC40305I        DSNUM = 1
BMC40091I 08/22/2014 17:17:14 SUBPHASE CHECKPOINT TAKEN FOR MERGE PHASE
BMC40012I MERGE PHASE COMPLETE. ELAPSED TIME = 00:00:25, TIME SINCE UTILITY START = 00:00:32
BMC40868I MERGE PHASE COMPLETE. ACCUMULATED TCB TIME = 00:00:00
Example 18: Using the MAXLSORT default value

Figure 132: Example 18--AFRPR002

BMC40018I RECOVER PLUS FOR DB2 V11.2.00
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884
7,769,718 AND 8,161,001
BMC40474I CONTACT BMC SUPPORT AT 1-800-537-1813 OR EMAIL TO SUPPORT@BMC.COM

BMC40016I MERGE PHASE STARTING 08/22/2014 17:16:49
BMC40301I TABLESPACE = G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2

BMC40484I MESSAGES FROM PHASE SUBTASK:

BMC40060I PENDING FLAG RECP SET FOR
BMC40345I SPACE = G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2
BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET
DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002
BMC40356I AN IDCAMS 'DEFINE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET
DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002
BMC40350I DATA SET DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002 WAS
SUCCESSFULLY ALLOCATED TO DDNAME SYS00006 ELAPSED = 00:00:00
BMC40417I 3 VSAM DATA SETS NOW ALLOCATED
BMC40485I END OF PHASE SUBTASK MESSAGES
BMC40350I DATA SET AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P02 WAS
SUCCESSFULLY ALLOCATED TO DDNAME SYS00011 ELAPSED = 00:00:00

BMC96304I THIS PHASE WILL RUN IN EXECUTION SUBTASK 2
BMC40593I START MERGE COMPLETE. ELAPSED TIME = 00:00:00
BMC40102I MERGE TABLESPACE STEP STARTING
BMC40795I FULL COPY DATASET IS AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P02
BMC40799I MERGE STARTING FOR DATA SET
DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002
BMC40593I INITIALIZATION COMPLETE, ELAPSED TIME = 00:00:00
BMC40593I SPIN COMPLETE, ELAPSED TIME = 00:00:00
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2
BMC40302I DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2
BMC40302I DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002
BMC40597I PROGRESS--PAGE NUMBER 100000, ELAPSED TIME = 00:00:11
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2
BMC40302I DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2
BMC40302I DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002
BMC40597I PROGRESS--PAGE NUMBER 200000, ELAPSED TIME = 00:00:21
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2
BMC40302I DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2
BMC40302I DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2
BMC40302I DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002
BMC40352I DDNAME SYS00006 (DSN
DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002) WAS SUCCESSFULLY UNALLOCATED
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2
BMC40302I DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002
BMC40599I PROGRESS--PAGE NUMBER 300000, ELAPSED TIME = 00:00:35
BMC40401I EXTENT ALLOCATED FOR TABLE SPACE G4GLMQDB.V9LMQTS1
BMC40305I DSNUM = 2
BMC40302I DSN = DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A002

Chapter 5 Examples of RECOVER PLUS jobs 459
Example 18: Using the MAXLSORT default value

Example 18: Using the MAXLSORT default value

Figure 133: Example 18--AFRPR003

BMC40018I RECOVER PLUS for DB2 V11.2.00
BMC40016I MERGE PHASE STARTING 08/22/2014 17:16:50
BMC40012I MERGE PHASE COMPLETE. ELAPSED TIME = 00:01:06, TIME SINCE UTILITY START = 00:01:14
BMC40060I PENDING FLAG RECP SET FOR SPACE = G4GLMQDB.V9LMQTS1
BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DEDLCAT.DSNDBC.G4GLMQDB.V9LMQTS1.I0001.A003
BMC40350I DATA SET DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A003 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00009 ELAPSED = 00:00:00
BMC40485I END OF PHASE SUBTASK MESSAGES
BMC40350I DATA SET AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P03 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00009 ELAPSED = 00:00:00
BMC96304I THIS PHASE WILL RUN IN EXECUTION SUBTASK 3
BMC40102I MERGE TABLESPACE STEP COMPLETE. ACCUMULATED TCB TIME = 00:00:00

BMC40018I RECOVER PLUS for DB2 V11.2.00
BMC40016I MERGE PHASE STARTING 08/22/2014 17:16:50
BMC40012I MERGE PHASE COMPLETE. ELAPSED TIME = 00:01:06, TIME SINCE UTILITY START = 00:01:14
BMC40060I PENDING FLAG RECP SET FOR SPACE = G4GLMQDB.V9LMQTS1
BMC40356I AN IDCAMS 'DELETE CLUSTER' OPERATION WAS SUCCESSFUL ON DATASET DEDLCAT.DSNDBC.G4GLMQDB.V9LMQTS1.I0001.A003
BMC40350I DATA SET DEDLCAT.DSNDBD.G4GLMQDB.V9LMQTS1.I0001.A003 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00009 ELAPSED = 00:00:00
BMC40485I END OF PHASE SUBTASK MESSAGES
BMC40350I DATA SET AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P03 WAS SUCCESSFULLY ALLOCATED TO DDNAME SYS00009 ELAPSED = 00:00:00
BMC96304I THIS PHASE WILL RUN IN EXECUTION SUBTASK 3
BMC40102I MERGE TABLESPACE STEP COMPLETE. ACCUMULATED TCB TIME = 00:00:00
Example 18: Using the MAXLSORT default value

Figure 134: Example 18 --AFROSUM
Example 18: Using the MAXLSORT default value

7,769,718 AND 8,161,001
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'0539' PSID = X'0002'
BMC40712I ORIGINAL STATUS = RO
BMC40716I STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000
BMC40723I ENCODING SCHEME = EBCDIC

BMC40820I TABLESPACE RECOVERY
BMC40827I LOG RESOURCE RANGE:
BMC40585I FROM LOGPOINT = X'000000000FB69FB3877' TO LOGPOINT =
\"X'000000000FB69942B835\"

BMC40840I INPUT COPY / SYSCOPY DATA:
BMC40640I 2014-08-22 17.14.20.6138 RBA/LRSN
X'000000000FB69FB3877' FULL IMAGE COPY
BMC40302I DSN = AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P01
BMC40641I SHRLEVEL = REFERENCE SITETYPE = LP DSNUM = 1
BMC40640I 2014-08-22 17.14.10.1773 RBA/LRSN
X'000000000FB69FB3877' QUIESCE UTILITY POINT - WRITE(YES)
BMC40640I 2014-08-22 17.14.20.6129 RBA/LRSN
X'000000000FB69907D252' -START ACCESS(FORCE)

BMC40709I OBJECT SUMMARY:
BMC40301I TABLESPACE = G4GLMQDB.V9LMQTS1
BMC40713I PARTITION = 2 OF 3
BMC40711I DBID = X'0539' PSID = X'0002'
BMC40712I ORIGINAL STATUS = RO
BMC40716I STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000
BMC40723I ENCODING SCHEME = EBCDIC

BMC40820I TABLESPACE RECOVERY
BMC40827I LOG RESOURCE RANGE:
BMC40585I FROM LOGPOINT = X'000000000FB69FB3877' TO LOGPOINT =
\"X'000000000FB69942B835\"

BMC40840I INPUT COPY / SYSCOPY DATA:
BMC40640I 2014-08-22 17.14.36.2125 RBA/LRSN
X'000000000FB69FB3877' FULL IMAGE COPY
BMC40302I DSN = AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P02
BMC40641I SHRLEVEL = REFERENCE SITETYPE = LP DSNUM = 2
BMC40640I 2014-08-22 17.14.10.1773 RBA/LRSN
X'000000000FB69FB3877' QUIESCE UTILITY POINT - WRITE(YES)
BMC40640I 2014-08-22 17.14.36.2118 RBA/LRSN
X'000000000FB699080DFB' -START ACCESS(FORCE)

BMC40709I OBJECT SUMMARY:
BMC40301I TABLESPACE = G4GLMQDB.V9LMQTS1
BMC40713I PARTITION = 3 OF 3
BMC40711I DBID = X'0539' PSID = X'0002'
BMC40712I ORIGINAL STATUS = RO
BMC40716I STOGROUP = G4GLMQSG PRIQTY = 25000 SECQTY = 50000
BMC40723I ENCODING SCHEME = EBCDIC

BMC40820I TABLESPACE RECOVERY
BMC40827I LOG RESOURCE RANGE:
BMC40585I FROM LOGPOINT = X'000000000FB69FB3877' TO LOGPOINT =
\"X'000000000FB69942B835\"

BMC40840I INPUT COPY / SYSCOPY DATA:
**Example 18: Using the MAXLSORT default value**

**Figure 135: Example 18--AFRPLAN**

BMC40018I RECOVER PLUS FOR DB2 V11.2.00  
BMC40473I COPYRIGHT BMC SOFTWARE INC. 1991-2015  
BMC96173I RECOVER PLUS TECHNOLOGY IS PROTECTED BY U.S. PATENTS 7,133,884  
7,769,718 AND 8,161,001  
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.BMCFCPYB.G4GLMQDB.V9LMQTS1.P01**  
**BMC40738I FILE OPTIMIZATION**  
**BMC40738I ALLOCATE 2 INPUT FILE BUFFER(S)**  
**BMC40738I OPEN IMAGE COPY FILE**  
**BMC40302I DSN = AFR.MXLTV9.BMCFCPYB.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**  
**BMC40345I SPACE = G4GLMQDB.V9LMQTS1**  
**BMC40305I DSNUM = 1**  
**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**
Example 18: Using the MAXLSORT default value

```
BMC40302I  DSN = AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P01
BMC40738I  FREE 2 INPUT FILE BUFFER(S)
BMC40738I  DEALLOCATE INPUT IMAGE COPY FILE
BMC40302I  DSN = AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P01
BMC40738I  RESET RECOVERY PENDING (RECP) FLAG
BMC40738I  REGISTER RECOVER TO CURRENT IN SYSCOPY
BMC40738I  RECORD RESTART INFORMATION

BMC40701I PHASE: MERGE
BMC40721I  PHASE INITIALIZATION:
BMC40301I   TABLESPACE = G4GLMQDB.V9LMQTS1
BMC40305I     DSNUM = 2
BMC40738I   ALLOCATE INPUT IMAGE COPY FILE
BMC40302I     DSN = AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P02
BMC40738I   FILE OPTIMIZATION
BMC40738I   ALLOCATE 2 INPUT FILE BUFFER(S)
BMC40738I   OPEN IMAGE COPY FILE
BMC40302I   DSN = AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P02
BMC40738I   ALLOCATE PAGE CHECK BUFFER (4096 BYTES)
BMC40738I   ALLOCATE SPACEMAP BUFFER (4096 BYTES)
BMC40738I   ALLOCATE PASTE BUFFER (4096 BYTES)
BMC40738I   ALLOCATE SPACE (READ/WRITE)
BMC40345I   SPACE = G4GLMQDB.V9LMQTS1
BMC40305I       DSNUM = 2
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 = 2
BMC40738I   PHASE TERMINATION:
BMC40738I   DISCONNECT FROM SPACE
BMC40738I   DEALLOCATE SPACE
BMC40345I   SPACE = G4GLMQDB.V9LMQTS1
BMC40305I       DSNUM = 2
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 = AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P02
BMC40738I   FREE 2 INPUT FILE BUFFER(S)
BMC40738I   DEALLOCATE INPUT IMAGE COPY FILE
BMC40302I   DSN = AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P02
BMC40738I   RESET RECOVERY PENDING (RECP) FLAG
BMC40738I   REGISTER RECOVER TO CURRENT IN SYSCOPY
BMC40738I   RECORD RESTART INFORMATION

BMC40701I PHASE: MERGE
BMC40721I  PHASE INITIALIZATION:
BMC40301I   TABLESPACE = G4GLMQDB.V9LMQTS1
BMC40305I     DSNUM = 3
BMC40738I   ALLOCATE INPUT IMAGE COPY FILE
BMC40302I     DSN = AFR.MXLTV9.BMCFCPYB.G4GLMQDB.V9LMQTS1.P03
BMC40738I   FILE OPTIMIZATION
BMC40738I   ALLOCATE 2 INPUT FILE BUFFER(S)
```
Example 18: Using the MAXLSORT default value

Chapter 5  Examples of RECOVER PLUS jobs  465
RECOVER PLUS concepts

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.
The following figure 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.

Figure 136: 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.

The following figure 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 137: 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 the following figure.

**Figure 138: 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. The following figure 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 139: 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 the following figure, 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 140: 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.
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:**
   ```sql
   RECOVER TABLESPACE X.Y DSNUM 1
   REBUILD INDEX (X.YCLUS) PART 1
   RECOVER UNLOADKEYS (X.YNCLUS) PART 1
   ```
   
   **JOB 2:**
   ```sql
   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:**
   ```sql
   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 81.
For more information, see “Understanding the LOGSORT strategy” on page 473 and “Understanding the UNLOADKEYS and BUILDINDEX strategy” on page 478.

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, 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 (limited by MAXDRIVES, see “MAXDRIVES integer ” on page 127), 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.

**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 “Understanding the NOWORKDDN strategy” on page 477), or the keys are written to the temporary data set specified by the WORKDDN parameter, usually SYSUT1 for 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 \textit{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 481.)

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 506.)

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 \texttt{R+/CHANGE ACCUM}. For more information about change accumulation files, see the \textit{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 \texttt{MAXLOGS} option. For more information about the additional resources that may be required for the LOGSORT strategy, see “Log sort space requirements” on page 340, “Managing sort performance” on page 489, and “Setting the SMCORE option” on page 493.

\textbf{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:

- \texttt{PIT\_RBA}: the point to which the table space is recovered

- \texttt{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 the following figure).

Figure 141: 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 the following figure), 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 142: When a recovery is not allowed by RECOVER PLUS

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 13 on page 343), the I/O reduction significantly reduces the
time that is required to rebuild the index.
If you do not specify WORKDDN, the NOWORKDDN option is the default. NOWORKDDN is required for the rebuild of the partitioning index when used together with the RECOVER UNLOADKEYS strategy for the nonpartitioning indexes.

For more information, see “NOWORKDDN” on page 236.

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 415.
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 197.

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 602). 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. The following figure 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.

Figure 143: BACKOUT recovery uses the spaces and logs

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:

RECOVER TABLESPACE PAYROLL.EMPLOYEE
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
— hash spaces

**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

LP Instant Snapshots are registered as IBM FlashCopy (FC) copies in SYSIBM.SYSCOPY. Other Instant Snapshots are registered in the BMC BMCXCOPY table with an STYPE of V. They are not registered in SYSIBM.SYSCOPY because the DB2 COPY utility does not recognize them. Standard DB2 utilities can only use FlashCopy (FC) copies registered in SYSIBM.SYSCOPY.

If a backup copy (LB or RB) is produced and its associated primary copy (LP or RP) is an Instant Snapshot, the backup is 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:

```
OPTIONS XBMID DBXM
RECOVER TABLESPACE PAYROLL.EMPLOYEE
   TABLESPACE PAYROLL.TAXRATES
   TABLESPACE PAYROLL.BENEFITS
   INDEXSPACE PAYROLL.EMPNUMIN
   INDEXSPACE PAYROLL.EMPNAMIN
   INDEXSPACE PAYROLL.TAXSTSN
   INDEXSPACE PAYROLL.BENEPLIN
```

Instant Snapshots and migration

When you use Instant Snapshots for migration and specify the OBIDXLAT option or when you specify the IMPORT or MIGRATE command, 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 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:

```
//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 64.

**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) (“RESOURCE SELECTION” on page 126). FlashCopy is also supported in the installation options LOCCPSEL (“LOCCPSEL=(FC,LP,LB)” on page 604) and REMCPSEL (“REMCSEL=(RP,RB,FC)” on page 605). 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 (“Non-registered copy or INLOG specification” on page 207).

The DSNAME description of the INCOPY specification accommodates FlashCopy support (“INCOPY specification” on page 217).

### Changing SHAREOPTIONS for striped or extended format data

RECOVER PLUS can read striped active logs or extended format active logs in non-data-sharing DB2 subsystems.

**Note**

If you do not have striped active logs or extended format logs, no action is required. You only need to change SHAREOPTIONS as described below if you use striped or extended format logs.

To allow RECOVER PLUS to read striped or extended format (usually greater than 2 GB) logs, you must define the logs with SHAREOPTIONS set to (2,3).

### 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.

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 473.

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, if the DB2 UT SORT DATA SET ALLOCATION field (UTSORTAL subsystem parameter) is set to NO, 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 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 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 497 and “Setting the MAXKSORT option” on page 495.

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 note 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.

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 497.
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 ("MAXLSORT=0" on page 606) or on the OPTIONS command ("MAXLSORT integer " on page 145), 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 495.) In this case, you use the KSORTSHARE option, which you can set in the installation options ("KSORTSHARE=YES" on page 607) or on the OPTIONS command ("KSORTSHARE " on page 146), 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 MAXSORT * 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 the following table.

<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.

See “SORT specification description” on page 124. The 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 338.

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 sections 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 “RECOVER PLUS syntax” on page 87.

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 “Examples of RECOVER PLUS jobs” on page 377. 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 504.

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
Allocating output image copy data sets dynamically

- 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 “Examples of RECOVER PLUS jobs” on page 377.)

**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
```
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 27 on page 511 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:

```
Incorrect: DSNAME &DB.&TS.&DATE
Correct: DSNAME &DB.&TS.RP&DATE
```

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).

### Stacking copies on tape

Text here
<table>
<thead>
<tr>
<th>Symbolic variable</th>
<th>Description</th>
<th>Length of result</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;JOBNAME</td>
<td>JOB name used in the JCL</td>
<td>8 bytes maximum</td>
</tr>
<tr>
<td>&amp;STEPNAM</td>
<td>STEP name used in the JCL (PROC names are ignored)</td>
<td>8 bytes maximum</td>
</tr>
<tr>
<td>&amp;VCAT</td>
<td>VCAT name for the space</td>
<td>8 bytes maximum</td>
</tr>
<tr>
<td>&amp;DB</td>
<td>database containing the space</td>
<td>8 bytes maximum</td>
</tr>
<tr>
<td>&amp;TS</td>
<td>table space or index space</td>
<td>8 bytes maximum</td>
</tr>
<tr>
<td>&amp;SP</td>
<td>table space or index space</td>
<td>8 bytes maximum</td>
</tr>
<tr>
<td>&amp;DSNUM, &amp;PART</td>
<td>data set or partition being copied</td>
<td>2 bytes for 0 - 99, 3 bytes for 100 - 999, 4 bytes for 1000 - 4096</td>
</tr>
<tr>
<td>&amp;LDSNUM, &amp;LPART</td>
<td>data set or partition being copied</td>
<td>3 bytes for 000-999, 4 bytes for 1000-4096</td>
</tr>
<tr>
<td>&amp;USERID or &amp;UID</td>
<td>job or TSO user ID</td>
<td>7 bytes maximum</td>
</tr>
<tr>
<td>&amp;SSID</td>
<td>DB2 subsystem ID</td>
<td>4 bytes</td>
</tr>
<tr>
<td>&amp;ATTACH</td>
<td>DB2 group attachment name or subsystem ID</td>
<td>4 bytes</td>
</tr>
<tr>
<td>&amp;DATE</td>
<td>current date (in the form YYMMDD)</td>
<td>6 bytes</td>
</tr>
<tr>
<td>&amp;TIME</td>
<td>current time (in the form HHMMS)</td>
<td>6 bytes</td>
</tr>
<tr>
<td>&amp;JDATE</td>
<td>current Julian date (in the form YYDDD)</td>
<td>5 bytes</td>
</tr>
<tr>
<td>&amp;YEAR</td>
<td>current year (in the form YY)</td>
<td>2 bytes</td>
</tr>
<tr>
<td>&amp;MONTH</td>
<td>current month (in the form MM)</td>
<td>2 bytes</td>
</tr>
<tr>
<td>&amp;DAY</td>
<td>current day (in the form DD)</td>
<td>2 bytes</td>
</tr>
<tr>
<td>&amp;JDAY</td>
<td>current Julian day (in the form DDD)</td>
<td>3 bytes</td>
</tr>
<tr>
<td>&amp;HOUR</td>
<td>current hour (in the form HH)</td>
<td>2 bytes</td>
</tr>
<tr>
<td>&amp;MINUTE</td>
<td>current minute (in the form MM)</td>
<td>2 bytes</td>
</tr>
<tr>
<td>&amp;SECOND</td>
<td>current second (in the form SS)</td>
<td>2 bytes</td>
</tr>
<tr>
<td>Symbolic variable</td>
<td>Description</td>
<td>Length of result</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
<td>-----------------</td>
</tr>
</tbody>
</table>
| &ICTYPE           | type of image copy:  
- F for FULL YES  
- I for FULL NO  
- A for FULL AUTO or CHANGELIMIT  
- D for FULL DSN1COPY | 1 byte |
| &UTIL             | utility ID | 8 bytes maximum |
| &SEQ              | a sequential number that restarts at 1 at the beginning of each job step execution | 2 bytes |
| &TYPE             | 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 | 2 bytes |
| &PART5            | partition for data set allocation  
You can use this variable for any data set.  
RECOVER PLUS generates 5-character partition numbers as follows:  
partition 1 = 00001  
partition 10 = 00010  
partition 100 = 00100  
partition 1000 = 01000  
nonpartitioned = 00000  
Example:  
ABC.DSN1.DA.&DB.&TSIX..P&PART5  
RECOVER PLUS generates the following 5-character partition number for partition 4096:  
ABC.DSN1.DA.DBNAME.TSNAME.P04096 | 5 bytes for table spaces with 4096 partitions or less |
<table>
<thead>
<tr>
<th>Symbolic variable</th>
<th>Description</th>
<th>Length of result</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;INST</td>
<td>instance number, with valid values of 1 or 2</td>
<td>1 bytes</td>
</tr>
<tr>
<td>&amp;UNIQ or &amp;UQ</td>
<td>1- to 8-character value, based on the system clock, that is used to generate unique copy data set names The first character is always an uppercase letter. Each remaining character is either an uppercase letter or a numeral from 0 through 9.</td>
<td>8 bytes maximum</td>
</tr>
</tbody>
</table>

1. Any trailing blanks in the result are removed.
2. The maximum total length allowed for a data set name is 44 bytes.
3. &TS for an index copy is the index space name. Using &TS is supported so that a single data set name can be specified for a group containing both table spaces and indexes.
4. Symbols with a numeric result must be prefixed by one or more alpha characters.
5. When used in an INDEP OUTSPACE MODEL, &DSNUM and &PART resolve to a 4-byte partition identifier for the data set node using the following scheme:
   - 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
6. This is the group attachment name if one is used as a RECOVER PLUS parameter; otherwise, the subsystem ID is used.
7. The values for these variables are assigned when the output copy data set is allocated.
8. Longer utility IDs are truncated to 8 characters.

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:

```plaintext
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
```

---

**Automatic sizing of output copies dynamically allocated on DASD**

You can use RECOVER PLUS to automatically size DASD output copies dynamically when the primary allocation quantity is not specified by the SPACE option.

Automatic sizing occurs if all of the following conditions exists:
You are using OUTCOPY (“OUTCOPY specification” on page 212).

The copy is dynamically allocated (using the OUTPUT command).

You did not specify the primary quantity with the SPACE option (“SPACE (primary, secondary) allocation unit” on page 157).

You specify either the AUTOSIZE=YES installation option (“AUTOSIZE=YES” on page 605) or YES for the AUTOSIZE option on the OPTIONS command (“AUTOSIZE” on page 147).

Optionally, you can use the MAXPRIM option on the OUTPUT command to set a maximum amount of disk space (in the units specified by SPACE) to allocate as primary space. Valid values are 0 through 65535. A nonzero value for MAXPRIM establishes an upper limit for primary space allocation, while a value of zero specifies no limit.

RECOVER PLUS calculates a value for the primary allocation and uses it if it is less than MAXPRIM. Then RECOVER PLUS calculates the secondary allocation as 20 percent of the primary allocation.

If the calculated value for the primary allocation exceeds MAXPRIM, RECOVER PLUS uses the MAXPRIM value. The secondary allocation is based on the remainder of (calculated primary allocation - MAXPRIM).

When automatic sizing occurs for OUTCOPY, the following message is in the MERGE phase output indicating the allocation values that RECOVER PLUS calculated:

BMC96009I ALLOCATING OUTPUT IMAGE COPY FILE WITH CYL(15,3)

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 versioned index spaces**

RECOVER PLUS supports the forward recovery of a versioned index.

Versioning occurs when the attributes of an index column have been 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).

- Page 4 will always be a directory page. If the index has been versioned several times, there may be more directory pages throughout the index.

BMC strongly recommends that you rebuild or REORG indexes with more than one directory page if you are creating image copies on a DB2 subsystem prior to DB2 Version 10. Otherwise, the recovery may fail. With DB2 Version 10, the directory pages will be written at the beginning of the image copy, insuring that the recovery will be successful.

With image copies made prior to DB2 Version 10, if a multi-data-set, nonpartitioned index contains a versioned column, 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 a versioned index, 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:

```sql
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.

The following table provides the columns in `SYSIBM.SYSTABLESPACESTATS1` and `SYSIBM.SYSINDEXSPACESTATS` that the RECOVER PLUS RECOVER command updates.

### Table 28: Columns updated by the RECOVER PLUS RECOVER command for real-time statistics

<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</td>
<td>timestamp</td>
</tr>
<tr>
<td>COPYUPDATEDPAGES</td>
<td>0</td>
</tr>
<tr>
<td>COPYCHANGES</td>
<td>0</td>
</tr>
<tr>
<td>COPYUPDATELRS</td>
<td>null</td>
</tr>
<tr>
<td>COPYUPDATETIME</td>
<td>null</td>
</tr>
</tbody>
</table>

1. 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, as well as the REBUILDLASTTIME column. The following table summarizes the support for updating these statistics when you use the RECOVER PLUS REBUILD command.

Table 29: 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>UPDATESTATSTIME</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>REORGDELETES</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>REORGEAFNEAR</td>
<td>0</td>
</tr>
<tr>
<td>REORGLEAFFAR</td>
<td>0</td>
</tr>
<tr>
<td>REORGNUMLEVELS</td>
<td>0</td>
</tr>
</tbody>
</table>

Recovering altered spaces

RECOVER PLUS supports online schema evolution, including recovering spaces that have had:

- partitions added or rotated
- their limit keys changed
- some materialized pending changes

Added partitions

If you add a partition with an ALTER statement or if a partition is automatically added with a partition-by-growth (PBG) 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.

When you are performing a drop recovery or a migration of a range-partitioned table space, the number of source partitions must match the target.

- For a DROPRECOVERY, IMPORT, or INCOPY request of PBG universal spaces where the number of source data sets differs from the number of target data sets, RECOVER PLUS determines the number of source data sets and acts accordingly.

- For a MIGRATE or INDEP OUTSPACE request, if the number of source data sets are greater than the number of target data sets, the request will fail. If the number of target is greater than the number of source data sets, the unused target data sets will be reset with a header page and space map.

### 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.

### Dropped partitions

If trailing partitions are removed from a PBG table space by a REORG on a subsystem where DSNZPARM parameter REORG_DROP_PBG_PARTS is enabled, you can perform the following recoveries with RECOVER PLUS:
■ You can recover the remaining partitions to the current time.

■ You can recover the remaining partitions to a point in time after the trailing partitions were removed.

You cannot recover table or index space partitions to a point in time before the partitions were removed.

## 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 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.

However, starting with DB2 Version 11, the change of limit keys for range-partitioned table is treated as pending change. In this case, the affected partitions are placed in AREOR status and are recorded in the SYSIBM.SYSPENDINGDDL DB2 catalog table.

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.

## Materialized pending changes

RECOVER PLUS allows you to recover to a point in time prior to a REORG that materialized some types of online pending changes.

This is allowable for UTS PBR, XML, and LOB spaces only. You can perform a point-in-time recovery over the following pending changes:

■ UTS PBR
  ■ DSSIZE
  ■ PGSIZE
  ■ SEGSIZE
  ■ MEMBER CLUSTER

■ XML auxiliary space
  ■ DSSIZE
—SEGSIZE

- LOB auxiliary space
- DSSIZE
- PGSIZE

The PIT recovery will set the table space in REORP status and insert a row into the SYSIBM.SYSPENDINGDDL DB2 catalog table. A REORG on the entire table space must be executed to complete the PIT recovery.

You cannot recover an index to a point in time that is prior to the materialization of pending definition changes.

**Note**
After executing REORG to materialize a DROP COLUMN pending change, you cannot perform a PIT recovery to a point prior to the materializing REORG.

---

**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) 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) 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
- for a *non-registered encrypted image copy*, use the ENCRYPTED option and TIMESTAMP option in the INCOPY specification of the RECOVER command (*“INCOPY specification” on page 217*)

**Note**

RECOVER PLUS finds encrypted copies registered in BMCXCOPY without the use of the syntax options.

- use the KEYDSNAM installation option to specify your key data set name (*“KEYDSNAM= keyDataSetName” on page 605*)
- create and maintain the key data set (*“Key data set” on page 522*)
- have the Recovery Management for DB2 solution and use a valid Recovery Management password

### 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 524. 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 (“KEYDSNAM= keyDataSetName” on page 605) 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>
</table>

An example of the contents of a key data set follows:

X'0ABCDEF123456789FEDCBA000111111' 2012-11-23-12-00  *128 bit DES encryption
X'123456789ABCDEF1'          2012-08-23-11-10
X'723DE6789000DEF1'  2011-12-16-16-40  DES *64 bit DES encryption
X'723DE6789000DEF1723DE6789000DEF1' 2011-12-12-14-00  AES *128 bit AES encrypt
X'F1F2F3F4F5F6F7F8'  2011-01-01-12-00

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 527.

For example, if RECOVER PLUS selected an image copy for a recovery from BMCXCOPY with a timestamp of 2011-02-12-10.00, 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” on page 525.) 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.

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.

Moving objects to another DB2 subsystem

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, check the SYSIBM.SYINDEXES catalog table on both subsystems to ensure that the value in both the CURRENT_VERSION column and the OLDEST_VERSION column is 0.

   Tip
   Ensure that you do an insert after the last ALTER to a table to force the creation of a system page, or use COPY PLUS with the GENSYSPAGES option to make the copy. If you are copying indexes and an index was altered, you may need to REBUILD the index to update the index directory page with the latest version before making the copy.

3. 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.
If you need to rebuild indexes on your target system, do not do so until after you have completed Step 4 on page 529.

4 Run DB2 REPAIR VERSIONS on the object on the target subsystem or specify the RECOVER PLUS UPDATE VERSIONS option in Step 3 on page 528.

This action updates the version numbers in the target system’s catalog using the version numbers in the system pages or index directory pages that have been laid down from the source 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:

\[
\begin{align*}
\text{CURRENT\_VERSION} &= \text{MAX}(\text{target.CURRENT\_VERSION}, \text{source.CURRENT\_VERSION}) \\
\text{OLDEST\_VERSION} &= \text{MIN}(\text{target.OLDEST\_VERSION}, \text{source.OLDEST\_VERSION})
\end{align*}
\]

5 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.

Use the CLONE option on the following RECOVER PLUS commands to recover with clone objects:

- RECOVER (“CLONE” on page 201)
- REBUILD (“CLONE” on page 235)
- RECOVER UNLOADKEYS (“CLONE” on page 250)
- RECOVER BUILINDEX (“CLONE” on page 258)
- LOGSCAN (“CLONE” on page 272)

When you are creating an output copy or a model DCB, you can use the symbolic variable &INST for the instance number (Table 27 on page 511). 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 its 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. 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

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.

Starting with DB2 Version 11, the following restriction applies to a recovering an XML object:
A point-in-time recovery can always convert an XML object from extended to basic RBA format. However, the point-in-time recovery for an XML table space cannot convert from basic to extended format.

Recovering compressed 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:

- IMPORT
- LOGSCAN
- MIGRATE
- RECOVER
- REBUILD INDEX
- SIMRCVR
- SIMBLD INDEX

For more information, see the syntax descriptions for each command in “RECOVER PLUS syntax” on page 87, and specifically, “OBJECTSET specification” on page 187.

Following are some examples of the use of the OBJECTSET 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 except for those table spaces named in the EXCLUDE statement |
| 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  
You can use the EXCLUDE statement to ignore one or more table spaces and their associated indexes. |
<p>| RECOVER INDEX OBJECTSET creator. name&lt;sup&gt;a&lt;/sup&gt; | recovers only indexes in the object set except for those indexes named in the EXCLUDE statement |
| REBUILD INDEX OBJECTSET creator. name&lt;sup&gt;b&lt;/sup&gt; | rebuilds all indexes in the object set except for those indexes named in the EXCLUDE statement |</p>
<table>
<thead>
<tr>
<th>Syntax</th>
<th>Objects copied</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGSCAN OBJECTSET <em>creator. name</em></td>
<td>log scan processing of all objects in the object set—table spaces and indexes</td>
</tr>
<tr>
<td>LOGSCAN TABLESPACE OBJECTSET <em>creator. name</em></td>
<td>log scan processing for only table spaces in the object set</td>
</tr>
<tr>
<td>LOGSCAN TABLESPACE OBJECTSET <em>creator. name</em> INDEX YES</td>
<td>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</td>
</tr>
<tr>
<td>LOGSCAN INDEX OBJECTSET <em>creator. name</em></td>
<td>log scan processing for only indexes in the object set</td>
</tr>
</tbody>
</table>

1  SIMRCVR is also valid.

2  SIMRBLD is also valid.

**Note**

DSNUM cannot be used with OBJECTSET. The DSNUM used for each object is its DSNUM in the repository.

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*.

## Using MIGRATE and IMPORT for data movement

With the Copy Migration feature of the Recovery Management solution, you can use the RECOVER PLUS MIGRATE and IMPORT commands to migrate a copy or set of copies to another DB2 subsystem.

The MIGRATE and IMPORT commands work with a sequential file created by the COPY PLUS EXPORT command. The sequential file contains BMCXCOPY and SYSCOPY table information about all selected table spaces. The RECOVER PLUS MIGRATE and IMPORT commands use the file to import migrated copies into the target location.
EXPORT, MIGRATE, and IMPORT make the task of moving the data between DB2 subsystems simpler and more accurate, and support all COPY PLUS formats (including cabinet copies).

If your migration uses the MIGRATE command, the COPY PLUS EXPORT runs on the target system and the RECOVER PLUS MIGRATE runs on the source system. RECOVER PLUS uses the OBIDs from the EXPORT. Optionally, log from the source system can be used with the MIGRATE command.

**Note**
The EXPORT file replaces the need for INCOPY and OBIDXLAT syntax with RECOVER PLUS. Records written to the file identify the correct data sets to use and how to translate the OBIDs from the source subsystem to the target subsystem.

This copy migration feature requires one of the following valid passwords:

- a Recovery Management solution password
- a Database Administration solution password

**Note**
For more information about the COPY PLUS EXPORT command, see the *COPY PLUS for DB2 Reference Manual.*

The BMC CHANGE MANAGER product also has a data migration function that handles DDL definition and OBID resolution. CHANGE MANAGER also uses the Copy Migration feature (EXPORT and IMPORT only, not MIGRATE) of COPY PLUS and RECOVER PLUS. For more information, see the CHANGE MANAGER product documentation.

The benefits of data migration with the COPY PLUS EXPORT and RECOVER PLUS MIGRATE/IMPORT feature include:

- supports importing copies that use BMC proprietary formats
- does not require that you manage data sets to keep up with current copies
- does not require that you know the OBIDs from the source system
- imports applications and object sets as a single unit
- does not require the import of unchanged objects
- supports the use of older EXPORT files to back-date imports
- provides a less error prone, and therefore, more accurate data migration (for example, less likely to use the wrong data set name or OBIDs).
If you use CHANGE MANAGER with the Copy Migration feature (EXPORT and IMPORT commands only, not the MIGRATE command), you achieve the following benefits:

- simplifies the data movement process
- requires less JCL, and therefore, less JCL management
- supports DSNUM values other than 0
- supports multiple imports and multitasking
- ability to use migration resources stored in BMCXCOPY, such as cabinet and OCC image copies

**COPY PLUS EXPORT file**

The COPY PLUS EXPORT file provides a method for transferring all types of COPY PLUS and IBM copies between DB2 subsystems.

The EXPORT file contains information to streamline the import recovery process with a minimum of user-supplied knowledge and intervention. The MIGRATE or IMPORT command of RECOVER PLUS is used to complete the transfer and integration of data using the copies.

The latest SYSCOPY and BMCXCOPY rows and metadata for the selected spaces are written to the designated sequential file. The metadata describes each object with information needed to translate OBIDs on the target system and to do checks during importing.

**Note**

EXPORT requires that the image copy is consistent to a point in time due to IMPORT processing on the target system done as TOCOPY. No log files are applied at the target. As the copies are selected for exporting, an attempt is made to verify that they are for the same consistent points.

If COPY PLUS finds an invalid image copy, COPY PLUS issues a message that indicates no copy was found. COPY PLUS assumes the EXPORT is for migration and continues processing. If a valid image copy is found, COPY PLUS issues a message that identifies the spaces that qualify for exporting.

The EXPOUT option on the COPY PLUS OUTPUT command indicates that COPY PLUS will create an EXPORT file for migration.
RECOVER PLUS MIGRATE and IMPORT commands

To initiate the import process, at a minimum you must specify the MIGRATE or IMPORT command with a USING clause. If table name ambiguity exists, you can add the CHANGE ... FROM ... TO clause. When all the tables on each target space match all the tables on one and only one space in the COPY PLUS EXPORT file, you do not need to specify the CHANGE ... FROM ... TO clause. Additionally, a SYNC option on the IMPORT command allows you to specify whether you want to migrate all spaces or only changed spaces.

For RECOVER PLUS MIGRATE and IMPORT command syntax, see “MIGRATE” on page 277 and “IMPORT” on page 297, respectively.

Limitations

The EXPORT/MIGRATE or IMPORT copy migration feature has the following limitations:

- no support for incremental copies
- no support for copies of the DB2 catalog
- no support for specifying DSNUM
  
  EXPORT looks for the most recent copy. If the copy is a DSNUM 0 copy, EXPORT uses that copy. If the copy is a DSNUM n copy, EXPORT looks for DSNUM n copies for all partitions or data sets.

- For IMPORT, RECOVER PLUS recovers indexes that are included in the IMPORT file and rebuilds other indexes, unless you specify INDEXES NO. If the source and target table spaces are not at the same version, RECOVER PLUS will not rebuild the indexes. You will need to run REPAIR VERSIONS or use the UPDATE VERSIONS option, and then rebuild the indexes in a separate step.
Recovering a dropped table space or table

This chapter provides an overview and examples for the recovery of a dropped table space or table.

Overview

The following considerations apply to drop recovery:

■ Drop recovery for indexes is not supported.

■ When you are performing a drop recovery for a range-partitioned table space, the number of partitions must match. For the drop recovery of a partition-by-growth (PBG) universal spaces where the number of source data sets differs from the number of target data sets, RECOVER PLUS determines the number of source data sets and acts accordingly.

■ 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 528.
The following figure shows how you can use the RECOVER PLUS product to recover a dropped table space.

**Figure 144: RECOVER PLUS drop recovery process**

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 545. 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 549.

### To recover a dropped table space when image copies are available

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 the following figure.

   **Figure 145: Example DDL for the dropped table space**

   ```sql
   CREATE TABLESPACE CRITTS01 IN XYZDB01
   FREEPAGE 0
   USING STOGROUP SYSDEFLT
   PRIQTY 40 SECQTY 40
   SEGSIZE 8;
   COMMIT;
   CREATE TABLE XYZDB01.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
   )
   IN XYZDB01.CRITTS01;
   COMMIT;
   CREATE UNIQUE INDEX XYZDB01.CRITIX01
   ON XYZDB01.CRITTB01 (EMPNO)
   USING STOGROUP SYSDEFLT PRIQTY 40 SECQTY 40 CLUSTER;
   COMMIT;
   ```

2. **Create DB2 objects.**
Using the DDL from Step 1 on page 541, 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 the following figure.

**Figure 146: Example JCL to run DSN1PRNT**

```plaintext
//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,                         <=
// IMAGE COPY DSN
// DISP=SHR
```

Output from this example job is shown in the following figure.

**Figure 147: Example DSN1PRNT output with the object identifiers of the dropped table space**

```
1DSN1999I START OF DSN1PRNT FOR JOB RDAMSMPR RUNPRNT
1DSN1998I INPUT DSNAME = RDAMSM.ICDBAJ.CRITTS01.G0003V00 , SEQ

HEADER PAGE: PGCOMB='10'X PLOGRBA='000000000000'X PLOGID='FF'X
PGNUMBER='000000'X
PFLAGS='38'X HPGBID='014C0002'X HPGRREF='00000001'X HPGRREL='C6'X
HPGTOBA='000000000000'X HPGTSTMP='00010101000000000000'X
HPGSNM='DBAJ'
HPGFID='0001'X HPGPSZ='1000'X HPGSGS='0008'X HPGPARTN='0000'X
HPGZPNUM='000000'X HPGZNUMB='00'X HPGTBLC='0001'X HPGROID='0003'X
HPGMAXL='0039'X
```
In the output from DSN1PRNT in the preceding example, 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 the preceding example). 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 the preceding example).

- **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 the preceding example).

**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 the following figure 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 148: Example SPUFI to locate new internal object identifiers

```sql
SELECT DBID,PSID
FROM SYSIBM.SYSTABLESPACE
WHERE DBNAME = 'XYZDB01'
    AND NAME   = 'CRITTS01';
```

<table>
<thead>
<tr>
<th>DBID</th>
<th>PSID</th>
</tr>
</thead>
<tbody>
<tr>
<td>332</td>
<td>12</td>
</tr>
</tbody>
</table>

DSNE610I NUMBER OF ROWS DISPLAYED IS 1

DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100

```sql
SELECT NAME,OBID
FROM SYSIBM.SYSTABLES
WHERE DBNAME = 'XYZDB01'
    AND TSNAME = 'CRITTS01';
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>OBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRITTB01</td>
<td>13</td>
</tr>
</tbody>
</table>

DSNE610I NUMBER OF ROWS DISPLAYED IS 1

DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100

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 147 on page 542.)

**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 the following figure.

**Figure 149: Example drop recovery RECOVER PLUS JCL**

```plaintext
/*===================================================================*/
/* 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                                                 
//        RDBMSM.ICDBAJ.CRITTS01.G0003V00                         
//        LOGPOINT X'0003082F5630'                              
//        OBIDXLAT DBID(X'014C',X'014C')                          
//        PSID(X'0002','X'000C')                                 
//        OBID(X'0003','X'000D')                                 
// 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.

**To recover a dropped table in a segmented table space containing multiple tables**

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 on page 546, 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.

3. Obtain internal object identifiers for 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 the following figure. Figure 151 on page 547 shows an example of DSN1PRNT output.

   The DBID, PSID, and OBIDs (except those for the original dropped table) are also available from the DB2 system catalog. See Step 4 on page 544 for more information about obtaining object identifiers from the catalog.

   **Figure 150: Example JCL to run DSN1PRNT to obtain object identifiers for original table space**

   ```bash
   //DSNO1 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=*  
   ```

546  RECOVER PLUS for DB2 Reference Manual
The first two bytes of `HPGOBID` are the hexadecimal representation of the DBID (X'014D' in the example output in the preceding example). The last two bytes of `HPGOBID` are the hexadecimal representation of the PSID of the table space (X'000A' in the example output in the preceding example).

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 the following figure.

**Figure 152: Example query for the DBID, PSID, and OBID values for the temporary table space**

```
SELECT A.DBNAME, A.TSNAME, A.NAME, HEX(A.DBID), HEX(B.PSID), HEX(A.OBID)
FROM SYSIBM.SYSTABLES A,
```
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 154 on page 550 and Figure 155 on page 550 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 the following figure.

Figure 153: Example RECOVER PLUS drop recovery JCL

```
//RECOVER  EXEC PGM=AFRMAIN,REGION=5M,
  PARM='DBA1,DMBBASIC,NEW,MSGLEVEL(1)' //STEPLIB  DD DISP=SHR,DSN=product.libraries
  DD DISP=SHR,DSN=DB2.DSNEXIT
  DD DISP=SHR,DSN=DB2.DSNLOAD
  DD * //SYSIN        DD *
  RECOVER TABLESPACE DMBDROPS.DMBSPAC1
    OBIDXLAT RESET
    OBID( X'14D', X'152') PSID( X'A', X'2')
    OBID( X'B', X'3') OBID( X'E', X'6') OBID( X'11', X'9')
    INDEP OUTSPACE
    MODEL DBAICAT.DSNDBC.DMBDROPS.DMBSPAC2.I0001.A001
    TORBA X'00031D928776'
```

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.
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.

To recover a dropped table from a simple table space

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 on page 549, 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.)
The following figure shows example JCL for the DSN1LOGP utility.

**Figure 154: Example DSN1LOGP utility JCL**

```sql
//DSN01 EXEC PGM=DSN1LOGP
//STEPLIB DD DSN=SYS2.DB2.PROD.DSNLOAD,DISP=SHR
// DD DSN=SYS2.DB2.PROD.DNSAMP,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(00039F08E8D2) DBID(0006) OBID(0009)
```

The following figure shows example DSN1LOGP utility output.

**Figure 155: Example DSN1LOGP utility output**

```
-- SEARCH --
CRITERIA
STARTRBA(00039F08E8D2) ENDRBA(FFFFFFFFFFFF)
SUMMARY(NO)
DATAONLY(NO)
SYSCOPY(NO)
ALL URIDS === YOU MAY SPECIFY URID(XXXXXXXXXXXXX)
ALL LUWIDS === YOU MAY SPECIFY
LUWID(NNNNNNNN.LLLLLLLL.XXXXXXXXXXXX.XXXX)
DBID(0006) OBID(0009)
ALL PAGES === YOU MAY SPECIFY PAGE(XXXXX) MANY TIMES
ALL TYPES === YOU MAY SPECIFY TYPE(XX)
ALL SUBTYPES === YOU MAY SPECIFY SUBTYPE(XX)
-- 33 --

DSN1212I DSN1LGRD FIRST LOG RBA ENCOUNTERED 00039F08E8D2
(LINES OMITTED)
00039F0AAB0E URID(00039F0A578A) LRSN(AE68AC13F86E) DBID(0006)
    OBID(0009) PAGE(000243) TYPE( UNDO  REDO )
    SUBTYPE(UPDATE IN-PLACE IN A DATA PAGE) CLR(NO)
    PROCNAME(DSN1LGBW)
    *LRH* 00A0D00D 06D18000 0E800000 C386CDD9 0000C386 D41E0126 000C386
    AC13F86E  * 8
0000
*LG** 10000600 09000243 27000000 C386D115
0000
  0000 00120000 0170900 0160000 00000000
4312

00039F0A959 URID(00039F0A578A) LRSN(AE68AC13F871) DBID(0006)
    OBID(0009) PAGE(000243) TYPE( UNDO  REDO )
    SUBTYPE(DELETE IN A DATA PAGE) CLR(NO)
    PROCNAME(DSN1LGBW)
    *LRH* 00D5004A 06D18000 0E800000 C386CDD9 0000C386 D49B0126 000C386
    D49BAE68  N  CF R CFM CFM
    AC13F871  * 8
0000
*LG** 10000600 09000243 39000000 C386D498
0000
  0000 009D2012 0170000 00009500 17120002 437C0002 437C0002 43140002
43130008  n
0020  C3D9C9E3 C9E7F0F2 E7E9E9C4 C2F0F140 0008C3D9 C9E3E3C2 F0F2E7EB
```
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 151 for details regarding the OUTPUT command and dynamic allocation of output data sets.) The following figure provides example RECOVER PLUS JCL for this step.

Figure 156: Example RECOVER PLUS JCL with dynamically allocated output copy

//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
  MODELCB SYS1.MODEL
  SPACE (20,1) CYL
RECOVER TABLESPACE DMBDROPR.DMBSPAC1
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.

**To recover a dropped table space using a pack backup**

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 the following figure.

   **Figure 157: Example DDL for the dropped table space**

   ```sql
   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 552, 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 the following figure uses the IBM utility DFDSS to restore the data sets.

**Figure 158: 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 HPGRBRBA value in the header page of the table space data set. The following figure shows the example RECOVER PLUS JCL.

**Figure 159: Example RECOVER PLUS JCL for recovery using a pack backup**

```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 *
RECOVER TABLESPACE ABCDB01.CRITTS01
   DROPRECOVERY
   OBIDXLAT DBID (X'014F',X'014F')
```
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.
Migrating data

This chapter provides examples for data migration using the RECOVER PLUS INCOPY and OBIDXLAT keywords (no Recovery Management solution password required).

Overview

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

If you own a Recovery solution, the migration process is simplified with the Copy Migration feature. Refer to the Recovery Management for DB2 User Guide for information on how to use the IMPORT (“IMPORT” on page 297) and MIGRATE (“MIGRATE” on page 277) commands. These commands eliminate the need to code the INCOPY and OBIDXLAT keywords.

Note

Online Consistent Copy 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 528.
The following figure 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 160: 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.)

The following figure 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.

Figure 161: RECOVER PLUS migration to a different complex

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 the migration of a partition-by-growth (PBG) universal spaces where the number of source data sets differs from the number of target data sets, RECOVER PLUS determines the number of source data sets and acts accordingly.

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).

If you have a Recovery Management solution password, CHANGE MANAGER supports the EXPORT and IMPORT commands (not MIGRATE) of the Copy Migration feature. For more information, see the CHANGE MANAGER user documentation and the *Recovery Management for DB2 User Guide*.

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 the following figure).

**Figure 162: Creating a consistent database for query and review**

If you have the BMC CHANGE MANAGER for DB2 product, you can use it to automate this entire process. CHANGE MANAGER generates the JCL and the syntax.

To create a database that is consistent for query and review, you perform the following steps.

**To create a consistent database for query and review**

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.

   **Note**
   
   See “Defining target objects for migration” on page 557 for more information.

   The DDL for this example is shown in the following figure.

   **Figure 163: Data definitions for source objects**

   ```sql
   CREATE DATABASE AFRORDER;
   CREATE TABLESPACE AFRORDER IN AFRORDER USING STOGROUP PROD1;
   CREATE TABLESPACE AFRSHIPS IN AFRORDER USING STOGROUP PROD1;
   CREATE TABLE AFRORDER
   (   ORDERNUM INT
   ```
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 the following figure. Also, consider other possible changes such as the STOGROUP and index type. The following figure shows the DDL for the target objects.

**Figure 164: Data definitions for target objects**

```sql
CREATE DATABASE AFRSUMOR;
CREATE TABLESPACE AFRSUMOR IN AFRSUMOR USING STOGROUP MIS1;
CREATE TABLESPACE AFRSUMSH IN AFRSUMOR USING STOGROUP MIS1;
CREATE TABLE AFRSUMORDER ( ORDERNUM INT
ITEMCODE INT
PRICE DECIMAL (5,2)
QUANTITY SMALLINT
TOTAL DECIMAL (7,2))
IN AFRSUMOR.AFRSUMOR;
CREATE TABLE AFRSUMSHIPS ( ORDERNUM INT
DATEORDERED DATE
DATESHIPPED DATE)
IN AFRSUMOR.AFRSUMSH;
CREATE UNIQUE INDEX AFRORDSI ON AFRSUMORDER (ORDENUM)
USING STOGROUP MIS1;
CREATE UNIQUE INDEX AFRSHPSI ON AFRSUMSHIPS (ORDENUM)
USING STOGROUP MIS1;
```

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 the following figure for an example.
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.

**Figure 165: Example SQL and output to select source and target objects**

```
SELECT NAME, DBID, PSID FROM SYSIBM.SYSTABLESPACE
WHERE NAME = 'AFRORDER' OR NAME = 'AFRSUMOR' OR NAME = 'AFRSHIPS' OR NAME = 'AFRSUMSH';
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>DBID</th>
<th>PSID</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFRORDER</td>
<td>352</td>
<td>2</td>
</tr>
<tr>
<td>AFRSHIPS</td>
<td>352</td>
<td>7</td>
</tr>
<tr>
<td>AFRSUMOR</td>
<td>449</td>
<td>2</td>
</tr>
<tr>
<td>AFRSUMSH</td>
<td>449</td>
<td>7</td>
</tr>
</tbody>
</table>

```
SELECT CREATOR, NAME, INDEXTYPE, DBID, ISOBID FROM SYSIBM.SYSINDEXES
WHERE NAME = 'AFRORDI' OR NAME = 'AFRSHPI' OR NAME = 'AFRORDSI' OR NAME = 'AFRSHPSI';
```

<table>
<thead>
<tr>
<th>CREATOR</th>
<th>NAME</th>
<th>INDEXTYPE</th>
<th>DBID</th>
<th>ISOBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>OWNERID</td>
<td>AFRORDI</td>
<td>2</td>
<td>352</td>
<td>5</td>
</tr>
<tr>
<td>OWNERID</td>
<td>AFRORDSI</td>
<td>2</td>
<td>352</td>
<td>5</td>
</tr>
<tr>
<td>OWNERID</td>
<td>AFRSHPI</td>
<td>2</td>
<td>449</td>
<td>10</td>
</tr>
<tr>
<td>OWNERID</td>
<td>AFRSHPSI</td>
<td>2</td>
<td>449</td>
<td>10</td>
</tr>
</tbody>
</table>

```
SELECT NAME, OBID FROM SYSIBM.SYSTABLES WHERE DBNAME = 'AFRORDER';
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>OBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFRORDER</td>
<td>3</td>
</tr>
<tr>
<td>AFRSHIPS</td>
<td>8</td>
</tr>
</tbody>
</table>

```
SELECT NAME, OBID FROM SYSIBM.SYSTABLES WHERE DBNAME = 'AFRSUMOR';
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>OBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFRSUMOR</td>
<td>3</td>
</tr>
<tr>
<td>AFRSUMSH</td>
<td>8</td>
</tr>
</tbody>
</table>

4 Quiesce the source objects.
Issue a QUIESCE command on the source table spaces. Example JCL and output are shown in the following figures.

**Figure 166: 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
QUIESCE TABLESPACE AFRORDER.AFRSHIPS
```

**Figure 167: Example QUIESCE output**

```
DSNU000I  DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = LSBRESMI
DSNU050I  DSNUGUTC - QUIESCE TABLESPACE AFRORDER.AFRORDER TABLESPACE AFRORDER.AFRSHIPS
DSNU477I *DSNP DSNQUIA - QUIESCE SUCCESSFUL FOR TABLESPACE AFRORDER.AFRORDER
DSNU477I *DSNP DSNQUIA - QUIESCE SUCCESSFUL FOR TABLESPACE AFRORDER.AFRSHIPS
DSNU474I *DSNP DSNQUIA - 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.

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 the following figure, 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.

**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 168: 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 (AFRSUMOR)
```

8 Allow queries.

Queries such as the one shown in the following figure (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 169: 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;
```

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>+----------------+----------------+</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>888.50</td>
</tr>
</tbody>
</table>

**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.

**To move an application to another subsystem with index recovery**

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 the following figure.

**Figure 170: 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 LASTSHUTUTDOWN 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. 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 “RECOVER PLUS syntax” on page 87.

In the example JCL shown in the following figure, you create migration images of the table spaces at the point of the pause of the DB2 subsystem that you captured in step 1. 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 “Making copies from the log and earlier copies” on page 581. You may combine these operations.

Note that AFRACCT is partitioned, but the OUTCOPY ASCODED option is used to make a copy that is for all of the partitions.

**Figure 171: Example RECOVER PLUS JCL to create migration images with index recovery**

```
//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,
```
3 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 557 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 572.
For this example, you would obtain the DDL shown in the following figure.

**Figure 172: Data definitions for source objects**

<table>
<thead>
<tr>
<th>SQL Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE DATABASE AFRFIN;</td>
<td>Create a database named AFRFIN.</td>
</tr>
<tr>
<td>CREATE TABLESPACE AFRACCT IN AFRFIN USING STOGROUP AFRTEST NUMPARTS 3;</td>
<td>Create a tablespace named AFRACCT.</td>
</tr>
<tr>
<td>CREATE TABLESPACE AFRTRAN IN AFRFIN USING STOGROUP AFRTEST;</td>
<td>Create a tablespace named AFRTRAN.</td>
</tr>
<tr>
<td>CREATE TABLESPACE AFRSUMM IN AFRFIN USING STOGROUP AFRTEST;</td>
<td>Create a tablespace named AFRSUMM.</td>
</tr>
<tr>
<td>CREATE TABLESPACE AFRHIST IN AFRFIN USING STOGROUP AFRTEST;</td>
<td>Create a tablespace named AFRHIST.</td>
</tr>
<tr>
<td>CREATE TABLE AFRACCT (ACCTNO INT, ACCTDESC CHAR(30), ACCTYPE CHAR(10), BALANCE DECIMAL (10,2)</td>
<td>Create a table named AFRACCT.</td>
</tr>
<tr>
<td>) IN AFRFIN.AFRACCT;</td>
<td></td>
</tr>
<tr>
<td>CREATE TABLE AFRTRAN (ACCTNO INT, TRANSDATE DATE, TRANSDESC CHAR(40), CLERKID CHAR(4) TOTAL</td>
<td>Create a table named AFRTRAN.</td>
</tr>
<tr>
<td>DECIMAL (10,2) ) IN AFRFIN.AFRTRAN;</td>
<td></td>
</tr>
<tr>
<td>CREATE TABLE AFRSUMM (ACCTNO INT, REPCAT CHAR(40), SUMMCAT CHAR(40), SUMMDATE DATE, TOTAL</td>
<td>Create a table named AFRSUMM.</td>
</tr>
<tr>
<td>DECIMAL (10,2) ) IN AFRFIN.AFRSUMM;</td>
<td></td>
</tr>
<tr>
<td>CREATE TABLE AFRHIST (ACCTNO INT, TRANSDATE DATE, TRANSDESC CHAR(40), CLERKID CHAR(4) TOTAL</td>
<td>Create a table named AFRHIST.</td>
</tr>
<tr>
<td>DECIMAL (10,2) ) IN AFRFIN.AFRHIST;</td>
<td></td>
</tr>
<tr>
<td>CREATE INDEX AFRACCTI ON AFRACCT (ACCTNO) USING STOGROUP AFRTEST CLUSTER (PART 1 VALUES(30000)</td>
<td>Create an index on AFRACCT.</td>
</tr>
<tr>
<td>,PART 2 VALUES(60000), PART 3 VALUES(99999) );</td>
<td></td>
</tr>
<tr>
<td>CREATE UNIQUE INDEX AFRTRANI ON AFRTRAN (ACCTNO) USING STOGROUP AFRTEST;</td>
<td>Create a unique index on AFRTRAN.</td>
</tr>
<tr>
<td>CREATE UNIQUE INDEX AFRSUMMI ON AFRSUMM (ACCTNO, REPCAT, SUMMCAT, SUMMDATE) USING STOGROUP</td>
<td>Create a unique index on AFRSUMM.</td>
</tr>
<tr>
<td>AFRTEST;</td>
<td></td>
</tr>
<tr>
<td>CREATE INDEX AFRHISTI ON AFRHIST (ACCTNO) USING STOGROUP AFRTEST;</td>
<td>Create an index on AFRHIST.</td>
</tr>
</tbody>
</table>

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 the following figure.
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 173: 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';
```

```
+----------------------------------+
<table>
<thead>
<tr>
<th>NAME</th>
<th>DBID</th>
<th>PSID</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFRACCT</td>
<td>321</td>
<td>2</td>
</tr>
<tr>
<td>AFRHIST</td>
<td>321</td>
<td>8</td>
</tr>
<tr>
<td>AFRSUMM</td>
<td>321</td>
<td>6</td>
</tr>
<tr>
<td>AFRTRAN</td>
<td>321</td>
<td>4</td>
</tr>
</tbody>
</table>
+----------------------------------+

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>CREATOR</th>
<th>NAME</th>
<th>INDEXTYPE</th>
<th>DBID</th>
<th>ISOBID</th>
<th>OBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>OWNERID</td>
<td>AFRACCTI</td>
<td>2</td>
<td>321</td>
<td>14</td>
<td>13</td>
</tr>
<tr>
<td>OWNERID</td>
<td>AFRHISTI</td>
<td>2</td>
<td>321</td>
<td>20</td>
<td>19</td>
</tr>
<tr>
<td>OWNERID</td>
<td>AFRSUMMI</td>
<td>2</td>
<td>321</td>
<td>18</td>
<td>17</td>
</tr>
<tr>
<td>OWNERID</td>
<td>AFRTRANI</td>
<td>2</td>
<td>321</td>
<td>16</td>
<td>15</td>
</tr>
</tbody>
</table>
+-----------------------------------------------------------------+

SELECT NAME, OBID FROM SYSIBM.SYSTABLES WHERE DBNAME = 'AFRFIN';
```

```
+--------------------------------+
<table>
<thead>
<tr>
<th>NAME</th>
<th>OBID</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFRACCT</td>
<td>9</td>
</tr>
<tr>
<td>AFRTRAN</td>
<td>10</td>
</tr>
<tr>
<td>AFRSUMM</td>
<td>11</td>
</tr>
<tr>
<td>AFRHIST</td>
<td>12</td>
</tr>
</tbody>
</table>
+--------------------------------+
```

5 Create objects on the target system.

Create the database, table spaces, tables, and indexes on the target system using the data definitions in the following figure. They need to be consistent in structure with the source system. (RECOVER PLUS provides migration checking when INCOPY is used.)

Figure 174: Data definitions for target objects

```
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
```
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 the following figure.

**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 175: Example SQL and output to obtain object IDs for target objects**

SELECT NAME, DBID, PSID FROM SYSIBM.SYSTABLESPACE WHERE
NAME = 'AFRTRAN' OR NAME = 'AFRACCT' OR
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. The ID translation is built from the data in step 4 and step 6. Output copies are made to allow a normal recovery on the audit system, if required.

---

**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 *Backup and Recovery Products for DB2 Messages* for message descriptions.
Example JCL for this process is shown in the following figure.

**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.

**Figure 176: Example RECOVER PLUS JCL to migrate data with index recovery**

```
//RECOOUTC1 EXEC PGM=AFRMAIN,REGION=0M,
// PARM='DBAJ,AFRAQDN,NEW,MSGLEVEL(1),,RDB2STAT(YES)'
//STEPLIB DD DISP=SHR,DSN=product.libraries
// DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPICK DD SYSOUT=* 
//ACCTCPY DD DSN=AFR.ACCCTCPYA,
// 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)
//SYSIN DD *
OPTIONS OUTCOPY ASCODED INDEXLOG YES
RECOVER TABLESPACE AFRAUDIT.AFRACCT
OBIDXLAT RESET
INCOPY 
    FULL DSNAME AFR.ACCTCPY1
    SHRLEVEL REFERENCE
    INVOLUME 311778
    INDEVT CART
    OUTCOPY YES OUTCOPYDDN(ACCTCPY)
    TOCOPY LASTCOPY
RECOVER TABLESPACE AFRAUDIT.AFRTRAN
OBIDXLAT RESET
INCOPY 
    FULL DSNAME AFR.TRANCPY1
    SHRLEVEL REFERENCE
    INVOLUME 311763
    INDEVT CART
    OUTCOPY YES OUTCOPYDDN(TRANCPY)
    TOCOPY LASTCOPY
RECOVER TABLESPACE AFRAUDIT.AFRSUMM
OBIDXLAT RESET
INCOPY 
    FULL DSNAME AFR.SUMMCPY1
    SHRLEVEL REFERENCE
    INVOLUME 311733
    INDEVT CART
    OUTCOPY YES OUTCOPYDDN(SUMMCPY)
    TOCOPY LASTCOPY
RECOVER TABLESPACE AFRAUDIT.AFRHIST
OBIDXLAT RESET
INCOPY 
    FULL DSNAME AFR.HISTCPY1
    SHRLEVEL REFERENCE
    INVOLUME 311722
    INDEVT CART
    OUTCOPY YES OUTCOPYDDN(HISTCPY)
    TOCOPY LASTCOPY
RECOVER INDEX OWNERID.AFRACCTI
OBIDXLAT RESET
```
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.
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.

To move an application to another subsystem with index rebuilding

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 the following figure.

Figure 177: 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

 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 558.) 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 “RECOVER PLUS syntax” on page 87.

In the example JCL shown in the following figure, you create migration images of the table spaces at the point of the pause of the DB2 subsystem that you captured.
in the previous step. 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 “Making copies from the log and earlier copies” on page 581.

Note that AFRACCT is partitioned, but the OUTCOPY ASCODED option is used to make a copy of all of the partitions.

Figure 178: Example RECOVER PLUS JCL to create migration images

```
//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
//SYSIPICK 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 *
//OPTIONS OUTCOPY ASCODED
RECOVER TABLESPACE AFRFIN.AFRACCT
OUTCOPY ONLY REGISTER NONE OUTCOPYDDN(ACCTCPY)
TABLESPACE AFRFIN.AFRTRAN
OUTCOPY ONLY REGISTER NONE OUTCOPYDDN(TRANCPY)
TABLESPACE AFRFIN.AFRSUMM
OUTCOPY ONLY REGISTER NONE OUTCOPYDDN(SUMMCPY)
TABLESPACE AFRFIN.AFRHIST
OUTCOPY ONLY REGISTER NONE OUTCOPYDDN(HISTCPY)
TORBA LASTSHUTDOWN
```

3 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 557 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 the following figure.

**Figure 179: Data definitions for the source objects**

```sql
CREATE DATABASE AFRFIN;
CREATE TABLESPACE AFRACCT IN AFRFIN USING STOGROUP AFRTEST
  NUMPARTS 3;
CREATE TABLESPACE AFRTRAN IN AFRFIN USING STOGROUP AFRTEST;
CREATE TABLESPACE AFRSUMM IN AFRFIN USING STOGROUP AFRTEST;
CREATE TABLESPACE AFRHIST IN AFRFIN USING STOGROUP AFRTEST;
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 the following figure.
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.

Figure 180: 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    |
|----------------------------------|
|1_ AFRACCT  |       321 |         2 |
|2_ AFRHIST  |       321 |         8 |
|3_ AFRSUMM  |       321 |         6 |
|4_ AFRTRAN  |       321 |         4 |
+----------------------------------+
```

```
SELECT NAME, OBID FROM SYSIBM.SYSTABLES WHERE DBNAME = 'AFRFIN';
```

```
+----------------------------------+
|        NAME        |   OBID    |
|---------------------|
|1_ AFRACCT           |         9 |
|2_ AFRTRAN           |        10 |
|3_ AFRSUMM           |        11 |
|4_ AFRHIST           |        12 |
+----------------------+
```

5 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.)

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 the following figure.

Figure 181: Data definitions for target objects

```
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
   (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 the following figure.

---

**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 182: Example SQL and output to obtain object IDs for target objects**

SELECT NAME, DBID, PSID FROM SYSIBM.SYSTABLESPACE WHERE 
NAME = 'AFRTRAN' OR NAME = 'AFRACCT' OR 
NAME = 'AFRHIST' OR NAME = 'AFRSUMM';
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 and step 6. Output copies are made to allow a normal recovery on the audit system, if required.

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 Backup and Recovery Products for DB2 Messages for message descriptions.

Example JCL for this process is shown in the following figure.

Figure 183: Example RECOVER PLUS JCL to migrate data with index rebuilding

//RECOUTC EXEC PGM=AFRMAIN,REGION=0M.
// 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
//SYSPICK DD SYSOUT=*  
//ACCTCPY 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)
Example 3: Moving an application to another subsystem with index rebuilding

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
Making copies from the log and earlier copies

This chapter provides examples of using the RECOVER PLUS product to create image copies.

Overview

You can use RECOVER PLUS to make updated image copies from typical recovery resources such as earlier copies, change accumulation files, and DB2 logs.

The following figure 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.

Figure 184: Making new copies from old copies and log records
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.

To create a consistent set of copies at a point in time

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 the following figure.

   Figure 185: Example JCL to QUIESCE a set of table spaces

   ```
   //QUIESCE1 EXEC DSNUPROC,SYSTEM=DBAJ,UID='AFRRM31',
   //  UTPROC='',REGION=4M,
   //  LIB='SYS2.DB2.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
   ```
Output from the QUIESCE follows in the following figure.

**Figure 186: Example QUIESCE output**

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSNU050I</td>
<td>DSNUGUTC - QUIESCE TABLESPACE AFRFIN.AFRACCT TABLESPACE AFRFIN.AFRTRAN TABLESPACE AFRFIN.AFRSUMM TABLESPACE AFRFIN.AFRHIST</td>
</tr>
<tr>
<td>DSNU477I &gt; DSNUQUIA</td>
<td>QUIESCE SUCCESSFUL FOR TABLESPACE AFRFIN.AFRACCT</td>
</tr>
<tr>
<td>DSNU477I &gt; DSNUQUIA</td>
<td>QUIESCE SUCCESSFUL FOR TABLESPACE AFRFIN.AFRTRAN</td>
</tr>
<tr>
<td>DSNU477I &gt; DSNUQUIA</td>
<td>QUIESCE SUCCESSFUL FOR TABLESPACE AFRFIN.AFRSUMM</td>
</tr>
<tr>
<td>DSNU477I &gt; DSNUQUIA</td>
<td>QUIESCE SUCCESSFUL FOR TABLESPACE AFRFIN.AFRHIST</td>
</tr>
<tr>
<td>DSNU474I &gt; DSNUQUIA</td>
<td>QUIESCE AT RBA 0004D176C05C</td>
</tr>
<tr>
<td>DSNU475I</td>
<td>DSNUQUIB - QUIESCE UTILITY COMPLETE, ELAPSED TIME = 00:00:00</td>
</tr>
<tr>
<td>DSNU010I</td>
<td>DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE = 0</td>
</tr>
</tbody>
</table>

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 the following figure 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 187: Example RECOVER PLUS JCL to create copies using dynamic allocation with automatic generation of GDGs**

```
//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
RECOVER TABLESPACE AFRIN.AFRACCT
   OUTCOPY ONLY   REGISTER ALL   OUTCOPYDDN COPY
INDEX ALL TABLESPACE AFRIN.AFRACCT
   OUTCOPY ONLY   REGISTER ALL   OUTCOPYDDN COPY
TABLESPACE AFRIN.AFRTRAN
   OUTCOPY ONLY   REGISTER ALL   OUTCOPYDDN COPY
INDEX ALL TABLESPACE AFRIN.AFRTRAN
   OUTCOPY ONLY   REGISTER ALL   OUTCOPYDDN COPY
TABLESPACE AFRIN.AFRSUMM
   OUTCOPY ONLY   REGISTER ALL   OUTCOPYDDN COPY
INDEX ALL TABLESPACE AFRIN.AFRSUMM
   OUTCOPY ONLY   REGISTER ALL   OUTCOPYDDN COPY
TABLESPACE AFRIN.AFRHIST
   OUTCOPY ONLY   REGISTER ALL   OUTCOPYDDN COPY
INDEX ALL TABLESPACE AFRIN.AFRHIST
   OUTCOPY ONLY   REGISTER ALL   OUTCOPYDDN COPY
TORBA LASTQUIESCE
```

**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 the following figure.

**Figure 188: Example RECOVER PLUS JCL to create new image copy using dynamic allocation**

```
//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 DSNAME AFR.IC.&DB.&SP(+1)
UNIT SYSDA CATLG YES MODELDCB SYS1.MODEL SPACE (10,10) CYL
RECOVER TABLESPACE AFRFIN.AFRACCT
   OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY
RECOVER TABLESPACE AFRFIN.AFRTRAN
   OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY
RECOVER TABLESPACE AFRFIN.AFRSUMM
   OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY
RECOVER TABLESPACE AFRFIN.AFRHIST
   OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY
INDEX AFRFIN.AFRACCTI
   OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY
INDEX AFRFIN.AFRTRANII
   OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY
INDEX AFRFIN.AFRSUMMI
   OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY
INDEX AFRFIN.AFRHISTII
   OUTCOPY ONLY REGISTER ALL OUTCOPYDDN COPY
```
Example 2: Creating an image copy without accessing the space
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 $C30DOPT job assembles the options macro. The macro contains the RECOVER PLUS processing options and the values for those options that are shipped with RECOVER PLUS. When the Installation System-generated customization job is submitted, it links the AFR$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 “Building and executing RECOVER PLUS jobs” on page 317. For more information about customizing your installation of RECOVER PLUS, see the Installation System Reference Manual and the BMC Products and Solutions for DB2 Customization Guide.
The following figure shows the installation options macro listing for RECOVER PLUS.

**Figure 189: RECOVER PLUS installation options module**

```sh
$AFROPTS &TBUFFS=1000, BUFFER MANAGER PAGES X
&PLANRECV=AFRBvu, RECOVER PLUS MAIN PLAN NAME X
&OPNDB2ID=YES, USE DB2 RACF ID X
&CHECKPT=PHASE, CHECKPOINT FOR RESTART X
&CHECKINT=0, INTERVAL BETWEEN CHECKPOINTS (MIN) X
&RCLTSK=10, MAXIMUM RECALL TASKS X
&AMSCAT=NO, USE CATALOG OPTION IN IDCAMS DEFINE X
&INDEXLOG=NO, SPECIFY RECOVER INDEX BEHAVIOR X
&IXRECP=NO, PUT INDEXES IN RECP STATUS if TS PIT RCVR X
&OUTCOPY=ASCODED, SPECIFY HOW TO CREATE OUTPUT IMAGE COPIES X
&SORTDEVT=, DEVICE TYPE FOR SORT WORK SPACE X
&SORTNUM=, NUMBER OF ALLOCATED SORT WORK SPACE DSNS X
&MXXSORT=, MAXIMUM NUMBER OF INDEX KEY SORTS X
&MXXLOGS=5, MAXIMUM LOGS TO ALLOCATE X
&MXXDRIVE=, MAXIMUM TAPE DRIVES TO BE USED X
&SMCORE=(0K,0K), MEMORY FOR SORT X
&RESINV=OK, MEMORY RESERVED FOR IDCAMS X
&WKUNIT=SYSALLDA, UNIT TO DYNAMICALLY ALLOCATE WORK FILES X
&XBMID=, XBM SUBSYSTEM ID OR XBM GROUP NAME X
&WTR=YES, WTOR FOR SPACES IN STOPP STATUS X
&ERRCONT=10, MAXIMUM TIMES TO CONT ON SEVERE ERROR X
&TZRULE=NONE, DAYLIGHT SAVINGS TIME (DST) SETTING X
&LOCCPSEL=(FC,LP,LB), LOCAL SITE COPY PRIMARY, SECONDARY X
&LOCCASEL=(LP,LB), LOCAL SITE CHANGE ACCUM PRIMARY,SECONDARY X
&REMPHASEL=(RP,RB,FC), REMOTE SITE COPY PRIMARY, SECONDARY X
&REMCASEL=(RP,RB), REMOTE SITE CHANGE ACCUM PRIMARY,SECONDARY X
&AUTOSIZE=YES, SIZE OUTPUT COPIES AND OUTPUT ACCUM FILES X
&CPUMIPS=200, RATE THAT CPU CAN EXECUTE INSTRUCTIONS X
&DISKIORATE=100, MEGABYTES PER SECOND READ FROM DISK X
&FCPPRC=NONE, Control FOR SNAP=VSAM WITH FLASHCOPY X
&SNAP=HW, RECOVER USING VSAM COPIES X
&USELOGS=(ACT1,ACT2,ARC1,ARC2) ORDER TO READ LOG FILES X
&KEYDSNAM=, NAME OF KEY DATA SET FOR ENCRYPTION X
&BINDQUALIFIER=AFRvvr, RECOVER PLUS BIND QUALIFIER X
&PUBLICPLAN=YES, RUN RECOVER PLUS WITH PUBLIC PRIVILEGE X
&DATAMVR=, PROGRAM TO USE FOR NONSNAPPABLE DASD X
&HISTORY=YES, BMC HISTORY TABLE X
&MXXSORT=0, NUMBER OF PARALLEL MERGES AND LOG SORTS X
&SHARESORT=YES, SH X
&AUX=NO, INCLUDE AUXILIARY OBJECTS IN RECOVERY X
&BACKOUT=AUTO, BACKOUT RECOVERY OPTION FOR PIT RECOVERY X
&AUSEHROBIDS=YES, USE OBIDS IN HEADER WHEN MIGRATING X
&ZIIP=ENABLED, ENABLE USE OF IBM SYSTEM ZIIPS X
&EATTR=, EXTENDED ATTRIBUTES SEQUENTIAL DATA SET X
&SSTORCLAS=, SMS STORAGE CLASS X
&DATACLAS=, SMS DATA CLASS X
&DEFINE=, &RICHK= X
```
RECOVER PLUS installation options

This section describes the RECOVER PLUS installation options including their default values.

For quick reference, the following table presents the options in alphabetical order and includes their default values, a brief description, and a reference to more information.

Table 30: RECOVER PLUS installation options

<table>
<thead>
<tr>
<th>RECOVER PLUS installation option</th>
<th>Default value</th>
<th>Brief description</th>
</tr>
</thead>
<tbody>
<tr>
<td>“AMSCAT=NO” on page 594</td>
<td>NO</td>
<td>defines use of catalog option in the IDCAMS define</td>
</tr>
<tr>
<td>AUOSIZE “AUTOSIZE=YES” on page 605</td>
<td>YES</td>
<td>specifies dynamic sizing for output image copies or change accumulation files</td>
</tr>
<tr>
<td>“AUX=NO” on page 608</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>
</tr>
<tr>
<td>“BACKOUT=AUTO” on page 609</td>
<td>AUTO</td>
<td>specifies the backout recovery option</td>
</tr>
<tr>
<td>“BINDQUALIFIER=AFRvvr” on page 606</td>
<td>AFRvvr</td>
<td>RECOVER PLUS bind qualifier, for example AFR111</td>
</tr>
<tr>
<td>“CHECKINT=0” on page 593</td>
<td>0</td>
<td>interval between checkpoints in minutes</td>
</tr>
<tr>
<td>“CHECKPT=PHASE” on page 593</td>
<td>PHASE</td>
<td>checkpoint for restart</td>
</tr>
<tr>
<td>“CPUMIPS=200” on page 613</td>
<td>200</td>
<td>rate that a CPU can execute instructions in millions of instructions per second (MIPS)</td>
</tr>
<tr>
<td>“DATACLAS” on page 612</td>
<td>none</td>
<td>SMS data class</td>
</tr>
<tr>
<td>“DATAMVR=” on page 606</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>
</tr>
<tr>
<td>“DEFINE=NO” on page 614</td>
<td>NO</td>
<td>specifies if DEFINE NO objects will be instantiated</td>
</tr>
<tr>
<td>RECOVER PLUS installation option</td>
<td>Default value</td>
<td>Brief description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>“DISKIORATE=100” on page 613</td>
<td>100</td>
<td>number of megabytes per second that RECOVER PLUS can read from disk</td>
</tr>
<tr>
<td>“EATTR=” on page 611</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>
</tr>
<tr>
<td>“ERRCONT=10” on page 602</td>
<td>10</td>
<td>maximum severe errors</td>
</tr>
<tr>
<td>“FCPPRC=NONE” on page 613</td>
<td>NONE</td>
<td>controls what happens if you specify SNAP=VSAM and the data sets are on disk that is capable of FlashCopy</td>
</tr>
<tr>
<td>“HISTORY=YES” on page 606</td>
<td>YES</td>
<td>use BMC HISTORY table</td>
</tr>
<tr>
<td>“INDEXLOG=NO” on page 594</td>
<td>NO</td>
<td>specifies RECOVER INDEX behavior</td>
</tr>
<tr>
<td>“IXRECP=NO” on page 595</td>
<td>NO</td>
<td>puts indexes in RECP status if TS PIT recovery</td>
</tr>
<tr>
<td>“KEYDSNAM=keyDataSetName” on page 605</td>
<td>none</td>
<td>name of the key data set for encryption</td>
</tr>
<tr>
<td>“KSORTSHARE=YES” on page 607</td>
<td>YES</td>
<td>specifies whether key sorts are shared, which determines number of active key sorts (with MAXKSORT)</td>
</tr>
<tr>
<td>“LOCCASEL=(LP, LB)” on page 604</td>
<td>(LP, LB)</td>
<td>resource selection for local site change accumulation files</td>
</tr>
<tr>
<td>“LOCCPSEL=(FC, LP, LB)” on page 604</td>
<td>(FC, LP, LB)</td>
<td>resource selection for local site copies</td>
</tr>
<tr>
<td>“MAXDRIVE=0” on page 599</td>
<td>0</td>
<td>specifies maximum tape drives to be used</td>
</tr>
<tr>
<td>“MAXKSORT” on page 597</td>
<td>min((2 * the number of CPUs), 12)</td>
<td>specifies the maximum number of index key sorts that can run concurrently</td>
</tr>
<tr>
<td>“MAXLOGS=5” on page 599</td>
<td>5</td>
<td>specifies the maximum logs to allocate</td>
</tr>
<tr>
<td>“MAXLSORT=0” on page 606</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>
</tr>
<tr>
<td>RECOVER PLUS installation option</td>
<td>Default value</td>
<td>Brief description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>“OPNDB2ID=YES” on page 593</td>
<td>YES</td>
<td>specifies the use of the DB2 RACF ID</td>
</tr>
<tr>
<td>“OUTCOPY=ASCODED” on page 596</td>
<td>ASCODED</td>
<td>specifies how to create output image copies</td>
</tr>
<tr>
<td>“PLANRECV=AFRBvvr ” on page 592</td>
<td>AFRB vvr</td>
<td>RECOVER PLUS main plan name, for example AFRB vvr</td>
</tr>
<tr>
<td>“PUBLICPLAN=YES” on page 606</td>
<td>YES</td>
<td>grants PUBLIC privilege to run RECOVER PLUS</td>
</tr>
<tr>
<td>“RCLTSK=10” on page 594</td>
<td>10</td>
<td>specifies the maximum recall tasks</td>
</tr>
<tr>
<td>“RDB2STAT=YES” on page 594</td>
<td>YES</td>
<td>restores original status of spaces</td>
</tr>
<tr>
<td>“REMCASEL=(RP,RB)” on page 605</td>
<td>(RP,RB)</td>
<td>resource selection for remote site change accumulation files</td>
</tr>
<tr>
<td>“REMCPSEL=(RP,RB,FC)” on page 605</td>
<td>(RP,RB,FC)</td>
<td>resource selection for remote site copies</td>
</tr>
<tr>
<td>“RESINV=0K” on page 600</td>
<td>0K</td>
<td>specifies memory reserves for IDCAMS</td>
</tr>
<tr>
<td>“RICHK=NO” on page 615</td>
<td>NO</td>
<td>specifies if RECOVER PLUS will check for referential integrity constraints on the objects being recovered</td>
</tr>
<tr>
<td>“SMCORE=(0K,0K)” on page 599</td>
<td>(0K,0K)</td>
<td>specifies memory for sort</td>
</tr>
<tr>
<td>“SNAP=HW” on page 612</td>
<td>HW</td>
<td>indicates if you want RECOVER PLUS to recover using VSAM copies, even if the data set is not on a snappable disk</td>
</tr>
<tr>
<td>“SORTDEVT” on page 596</td>
<td>determined by BMCSORT installation options</td>
<td>specifies device type for sort work space</td>
</tr>
<tr>
<td>“SORTNUM” on page 597</td>
<td>determined by BMCSORT installation options</td>
<td>specifies number of allocated sort work space DSNs</td>
</tr>
<tr>
<td>“STORCLAS” on page 612</td>
<td>none</td>
<td>SMS storage class</td>
</tr>
<tr>
<td>“TBUFFS=1000” on page 592</td>
<td>1000</td>
<td>specifies buffer manager pages</td>
</tr>
<tr>
<td>“TZRULE=NONE” on page 602</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>
</tr>
<tr>
<td>“USEHDROBIDS=YES” on page 610</td>
<td>YES</td>
<td>indicates if the OBIDs in the header are valid or not.</td>
</tr>
</tbody>
</table>
## RECOVER PLUS installation options

<table>
<thead>
<tr>
<th>RECOVER PLUS installation option</th>
<th>Default value</th>
<th>Brief description</th>
</tr>
</thead>
<tbody>
<tr>
<td>“USELOGS=(ACT1,ACT2,ARC1,ARC2)” on page 614</td>
<td>(ACT1,ACT2,ARC1,ARC2)</td>
<td>specifies the order in which to read active and archive logs</td>
</tr>
<tr>
<td>“WKUNIT=SYSALLDA” on page 600</td>
<td>SYSALLDA</td>
<td>indicates unit to dynamically allocate work files</td>
</tr>
<tr>
<td>“WTOR=YES” on page 602</td>
<td>YES</td>
<td>indicates to issue WTOR if RECOVER PLUS finds spaces in STOPP status</td>
</tr>
<tr>
<td>“XBMID=ssid or xbmGroup” on page 600</td>
<td>none</td>
<td>specifies the XBM subsystem ID or XBM group name for Instant Snapshots</td>
</tr>
<tr>
<td>“ZIIP=ENABLED” on page 601</td>
<td>ENABLED</td>
<td>determines whether zIIP processing is enabled</td>
</tr>
</tbody>
</table>

1 The default values for the installation options are those generated through the Installation System. The installation system enables you to customize the default values for your site and saves them in the installation system profile variables.

Descriptions of each of the installation options with their default values follow.

### Basic installation options

This section describes the basic RECOVER PLUS installation options and their default values.

**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=AFRB ννr**

This option is the RECOVER PLUS execution plan. The default value is AFRB ννr, where νν represents a two-character version number and r represents a one-character release number, such as AFRB11.
**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.

**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 “Checkpoint override parameter (checkPt)” on page 322). 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 “CHECKINT (integer)” on page 122).

**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 “RDB2STAT override parameter (rdb2Stat)” on page 323).

**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. RCLTSK=0 disables EARLYRECALL. The default value is 10.

For more information about EARLYRECALL, see “EARLYRECALL / NOEARLYRECALL” on page 118.

**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
INDEXLOG=NO 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.

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.

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
You can override the IXRECP installation option by using the IXRECP keyword with the OPTIONS command (see “IXRECP” on page 135).

**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 “OUTCOPY” on page 127).

**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 94.
**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 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.

For more information, see “General information about SORTDEVT and SORTNUM” on page 94.

**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 607.)
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:

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

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. For more information, see “Understanding the UNLOADKEYS and BUILDINDEX strategy” on page 478.

When you specify MAXKSORT, the following files are dynamically allocated if you do not code them in JCL:

- **SYSO\*nnn**: sort message files
  
  \(*nnn\) is a number between 1 and the value that is specified for MAXKSORT.

- **S\*xxxWKnn**: 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.

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 235.
**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 “MAXLOGS integer ” on page 127).

**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 “MAXDRIVES integer ” on page 127).

**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 489.

- 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 (“ZIIP” on page 137 and “ZIIP=ENABLED” on page 601) 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 “XBMID ssid or xbmGroup” on page 136).

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 600 or “XBMID ssid or xbmGroup” on page 136.

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 ("ZIIP" on page 137).

**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=None**

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=startTimeRule[/time].endTimeRule[/time]
```

The first date (startTimeRule) describes when the change to DST occurs, and the second date (endTimeRule) 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/M11.1.0
  ```

  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
  ```

- **n**

  Specify the zero-based Julian day `n`, where `n` is a value from 0 through 365. In this format, leap days are counted, and you can refer to February 29.
  
  In the following example, DST begins on April 1 (leap years), or April 2 (nonleap years) at 12:34:56 p.m., and ends on November 1 at 2:34:56 a.m.:

  ```
  TZRULE=91/12:34:56,305/2:34:56
  ```

**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:nn.

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:nn. 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).

**LOCPSEL=(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, system backup (SB), 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.
**REMCSEL=(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 output image copies or 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 output image copies or change accumulation output files to DASD occurs. AUTOSIZE=NO means output image copies or 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 “AUTOSIZE” on page 147).

**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 521.

---

**Note**

Encryption is a feature of the Recovery Management for DB2 solution and requires a valid Recovery Management solution password.
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 “DATAMVR programName” on page 138).

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 AFR111.

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 “BMCHIST table” on page 628 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.

When you specify MAXLSORT, the following files are dynamically allocated if you do not code them in JCL:

- LOGOU*n*nn: 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.

- LxxxWK*n*n: sort work files
  
  *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. For example, if MAXLSORT=3 and two sort work files are required for each sort, the DDs would be specified as follows:

```plaintext
L001WK01 DD...
L001WK02 DD...
L002WK01 DD...
L002WK02 DD...
L003WK01 DD...
L003WK02 DD...
```

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

**KSORTSHARE=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 “KSORTSHARE” on page 146).

**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, ALL, YES, XML, LOB, and HISTORY. The default value is NO.

These values have the following meanings:

- **AUX=NO**: Does not include any auxiliary objects in the recovery.

- **AUX=ALL**: Includes XML and LOB objects in the recovery. It does not include HISTORY spaces with their related auxiliary objects when a table space with system-maintained temporal tables is specified in SYSIN.

- **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 ("AUX" on page 128) or in the table space specification for any of the following RECOVER PLUS statements:

- RECOVER TABLESPACE
- RECOVER INDEX(ALL) TABLESPACE
- RECOVER INDEXSPACE(ALL) TABLESPACE
- REBUILD INDEX(ALL) TABLESPACE
- LOGSCAN TABLESPACE
- LOGSCAN INDEX(ALL) TABLESPACE
- LOGSCAN INDEXSPACE(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=ALL, 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 ("BACKOUT" on page 131).

For more information about BACKOUT AUTO recovery, see "Using BACKOUT AUTO recovery" on page 50.

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 ("OBIDXLAT" on page 206) 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 "USEHDROBIDS" on page 130).

**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).

You can set EATTR to OPT or NO in the JCL.

When you create an OUTCOPY (“OUTCOPY specification” on page 212) and want to force the copy to the cylinder-managed space of an EAV, specify the SPACE keyword.

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=YES, 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 “EATTR” on page 158.)

**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.

**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.

**SNAP=HW**

Use SNAP to indicate if you want RECOVER PLUS to read VSAM copies, even if the data set is not on a snappable disk:

- SNAP=HW (the default) tells RECOVER PLUS to use a hardware data set snapshot to restore an Instant Snapshot or VSAM data set.
SNAP=VSAM tells RECOVER PLUS to use conventional VSAM I/O to restore a VSAM data set if it is not on a snappable disk.

To read a VSAM copy with SNAP VSAM, you specify the name of the VSAM data set in an INCOPY statement, just as you would if it was an Instant Snapshot copy, using the INCOPY FULL SNAPSHOT DSNAME dataSetName syntax.

SNAP VSAM also allows you to recover using a VSAM copy registered in BMCXCOPY or SYSCOPY if that copy is not on snappable disk.

You can override the value of the SNAP installation option at runtime by specifying the SNAP option on the OPTIONS command (“SNAP” on page 151).

**FCPPRC=NONE**

Use FCPPRC to control what happens if you specify SNAP=VSAM, and the data sets are on disk that is capable of FlashCopy.

The following values are valid for the FCPPRC installation option when the target volume is a primary in a PPRC mirror pair:

- NONE (the default) specifies that no FCTOPPRCPRIORITY option is passed to DFDSS.
- PREF specifies that preserving the mirror during the FlashCopy is preferred.
- REQ specifies that preserving the mirror is required.
- NO specifies that the mirror is not preserved during a FlashCopy.

**DISKIORATE=100**

Use DISKIORATE to specify the number of megabytes per second that RECOVER PLUS can read from disk. The default value is 100.

*Note*

DISKIORATE is used to derive many of the performance factors for estimation. Estimation requires a Recovery Management for DB2 solution password. For more information about estimation, see Chapter 8, "Recovery simulation and estimation," in the *Recovery Management for DB2 User Guide.*

**CPUMIPS=200**

Use CPUMIPS to indicate the rate that a CPU can execute instructions in millions of instructions per second (MIPS). The default value is 200.
Note

CPUMIPS is used to derive many of the performance factors for estimation. Estimation requires a Recovery Management for DB2 solution password. For more information about estimation, see Chapter 8, "Recovery simulation and estimation," in the Recovery Management for DB2 User Guide.

USELOGS=(ACT1,ACT2,ARC1,ARC2)

Use USELOGS to specify the order in which to read active and archive logs. The default is (ACT1,ACT2,ARC1,ARC2).

You can override the USELOGS value at runtime by specifying the RESOURCE SELECTION LOGS syntax on the OPTION command ("RESOURCE SELECTION" on page 126).

Values for USELOGS are as follows:

- **ACT1**
  Indicates the primary active log file.

- **ACT2**
  Indicates the dual (secondary) active log file.

- **ARC1**
  Indicates the primary archive log file.

- **ARC2**
  Indicates the dual (secondary) archive log file.

DEFINE=NO

The DEFINE option specifies if DEFINE NO objects will be instantiated. Valid values are YES and NO. The default value is NO. If you specify YES, RECOVER PLUS instantiates DEFINE NO objects if necessary, and updates the catalog and DBD to reflect that the object has been defined. If you specify NO, RECOVER PLUS does not process DEFINE NO objects.

You can override the DEFINE installation option by using the DEFINE keyword with the OPTIONS command (see “DEFINE” on page 131) or in the options for the RECOVER or IMPORT commands.
RICHK=NO

The RICHK option specifies if RECOVER PLUS will check for referential integrity constraints on the objects being recovered. Valid values are YES and NO. The default value is NO. If you specify YES, RECOVER PLUS checks for referential integrity constraints and sets check pending (CHKP) status for dependent objects not recovered in the same step to the same PIT. If you specify NO, RECOVER PLUS does not check for referential integrity.

You can override the RICHK installation option by specifying the RICHK option in the OPTIONS command (see “RICHK” on page 128) or in the options for the RECOVER or IMPORT commands.
BMCSORT installation and options

This appendix describes

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” on page 617.

DYNALOC installation option

The DYNALOC installation option of BMCSORT provides information for dynamically allocating SORTWK data sets.

BMCSORT deallocated these data sets at the end of each sort. The content of the $AUPSMAC macro in $532SOPT 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
```

DYNAMIC ALLOC OPTIONS FOR SORT

The values that you specify in this macro apply to all invocations of BMCSORT. BMCSORT uses the same options module for all BMC products that invoke BMCSORT.

Table 31 on page 618 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 also override the BMCSORT options values that you specify when one of the following conditions exists:

- The values in the invoking product’s dynamic allocation installation options or corresponding command options conflict with the values that you specify.
- You turn on BMCSORT SORTWK dynamic allocation from the product that invokes BMCSORT, and you specify OFF for the position 3 parameter.

BMCSORT dynamically allocates SORTWK files as necessary.

### Table 31: DYNALOC parameters

<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 specifies the generic unit name from which the BMC product 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. The BMC product does not use this parameter, but the parameter is required for proper assembly of the installation options macro.</td>
<td>3</td>
<td>Do not change this value.</td>
</tr>
</tbody>
</table>
| Position 3                | This parameter tells the BMC product whether to dynamically allocate SORTWK files. BMC recommends that you not change this value. | ON | - ON dynamically allocates SORTWK.  
- OFF does not dynamically allocate SORTWK. |
<p>| Position 4                | Do not change this value. The BMC product does not use this parameter, but the parameter is required for proper assembly of the installation options macro. | ON | Do not change this value. |</p>
<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 5</td>
<td>Do not change this value. The BMC product does not use this parameter, but the parameter is required for proper assembly of the installation options macro.</td>
<td>6000000</td>
<td>Do not change this value.</td>
</tr>
<tr>
<td>Position 6</td>
<td>Do not change this value. The BMC product does not use this parameter, but the parameter is required for proper assembly of the installation options macro.</td>
<td>3000000</td>
<td>Do not change this value.</td>
</tr>
</tbody>
</table>
| Position 7                | This parameter specifies the DASD type with the smallest track capacity that a dynamically allocated SORTWK data set might encounter at your site. | 3390 | ■ 3380, track capacity of 47968  
■ 3390, track capacity of 56664  
■ 9345, track capacity of 46456 |
| 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, the BMC product 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. |
<table>
<thead>
<tr>
<th>Parameter name or position</th>
<th>Description</th>
<th>Initial value</th>
<th>Valid values</th>
</tr>
</thead>
</table>
| RETRY                     | This parameter specifies how you want the BMC product to handle retry attempts for SORTWK dynamic allocation:  
  - The first subparameter indicates the number of times that you want the BMC product 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 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 rely on the retry function, BMC recommends that you use the same values as your SyncSort RETRY installation parameter. | (0,0) | If you use this parameter, BMC recommends that you specify the same values as your SyncSort RETRY installation parameter. The following values are valid for this parameter:  
  - 0 through 16 for the first subparameter  
    0 indicates that you do not want the BMC product to retry the request.  
  - 0 through 15 for the second subparameter  
    0 indicates that you do not want the BMC product to retry the request. |
Common utility tables

This chapter describes the contents of the common utility tables, considerations for these tables, and how to maintain them if necessary.

Overview of common utility tables

The BMC common utility tables contain information about the BMC utilities that you generate and submit through a BMC utility product.

Table 32 on page 621 lists the tables that each utility uses and each table’s default name and synonym.

Table 32: Common utility tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Default name</th>
<th>Synonym</th>
<th>Utilities that use this table</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMCDICT</td>
<td>CMN_BMCDICT</td>
<td>BMC_BMCDICT</td>
<td>■ LOADPLUS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ REORG PLUS</td>
</tr>
<tr>
<td>BMCHIST</td>
<td>CMN_BMCHIST</td>
<td>BMC_BMCHIST</td>
<td>■ CHECK PLUS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ COPY PLUS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ LOADPLUS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ RECOVER PLUS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ REORG PLUS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ UNLOAD PLUS</td>
</tr>
<tr>
<td>Table</td>
<td>Default name</td>
<td>Synonym</td>
<td>Utilities that use this table</td>
</tr>
<tr>
<td>-----------</td>
<td>--------------</td>
<td>-------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>BMCLGRNX</td>
<td>CMN_BMCLGRNX</td>
<td>BMC_BMCLGRNX</td>
<td>COPY PLUS, Log Master, RECOVER PLUS, RECOVERY MANAGER</td>
</tr>
<tr>
<td>BMCSYNC</td>
<td>CMN_BMCSYNC</td>
<td>BMC_BMCSYNC</td>
<td>CHECK PLUS, COPY PLUS, DASD MANAGER PLUS (BMCSTATS), LOADPLUS, RECOVER PLUS, RECOVERY MANAGER, REORG PLUS, UNLOAD PLUS</td>
</tr>
<tr>
<td>BMCTRANS</td>
<td>CMN_BMCTRANS</td>
<td>BMC_BMCTRANS</td>
<td>Log Master, RECOVERY MANAGER</td>
</tr>
</tbody>
</table>
## Warnings and considerations for common utility tables

This topic describes important information that you need to know when using the common utility tables.
**WARNING**

The following warnings apply to the common utility tables:

- 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 any BMC table except the BMCHIST table.

- 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.

Note the following considerations:

- 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
  - BMCXCOPY

- Supported versions of the following BMC products support the LOCKROW installation option:
  - CHECK PLUS
  - LOADPLUS
  - REORG PLUS
If the value of the option is YES (which is the default value), the products use MVS enqueues instead of SQL LOCK TABLE statements to serialize updates to the BMCSYNC and BMCUTIL tables. The following BMC products always use MVS enqueues for serialization when updating the BMCSYNC and BMCUTIL tables:

— COPY PLUS
— DASD MANAGER PLUS
— RECOVER PLUS

Managing common utility tables

This topic 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.

1 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.

  ■ Run the following SQL statement, replacing *tableName* with a BMC common utility table name (listed in “Overview of common utility tables” on page 621):

    ```sql
    SELECT CREATOR, NAME FROM SYSIBM.SYSTABLES
    WHERE TSNAME='*tableName*';
    ```

  ■ Get the names from your DB2 system administrator.

To query the tables

1 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:

```sql
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:

```sql
SELECT DBNAME, SPNAME, CHAR(ELAPSED, ISO), CHAR(TIME, ISO)
FROM creatorName.CMN_BMCHIST
WHERE UTILID='utilityID';
```

### To display BMC utility status

1. 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:

```sql
SELECT * FROM creatorName.CMN_BMCUTIL
WHERE DBNAME='databaseName'
AND SPNAME='tableSpaceName'
```

```sql
SELECT * FROM creatorName.CMN_BMCSYNC
WHERE NAME1='databaseName'
AND NAME2='spaceName';
```

### To terminate a BMC utility

1. To terminate a BMC utility, perform one of the following actions:

   - To terminate a BMC utility that is executing, use the following SQL statements:

     ```sql
     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 the TERM restart parameter.

---

**BMCDICT table**

The BMCDICT table stores the compression dictionary during load or reorganization processing.

Table 33 on page 627 describes the contents of the BMCDICT table.
Table 33: Contents of the 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>

BMCDICT table considerations

This topic describes important information that you need to know about the BMCDICT table:

- 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 or reorganizing with REORG PLUS).

- 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.

On rare occasions, you might need to take action to control expansion of the BMCDICT table.

**To control expansion of the BMCDICT table**

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:

```
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**

The BMCHIST table contains information about completed executions of the BMC utilities for DB2.

The following configuration or 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 a configuration or installation option), the utility inserts rows into the BMCHIST table during the UTILTERM phase.

For COPY PLUS, if the value is SUMMARY, the utility inserts only summary information about the COPY PLUS execution into the BMCHIST table. This option provides less information than the YES option.

*Table 34 on page 628* describes the contents of the BMCHIST table.

**Table 34: Contents of the 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>Column name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>----------------</td>
<td>----------------------------------------------------------------------------</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>■ RECOVER</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 (COPY PLUS)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>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></td>
<td></td>
<td>■ For UNLOAD PLUS, if you specified LOGICAL PART, these partitions are the physical partitions that correspond to the logical partitions that you specified.</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>Column name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------</td>
<td>--------------------------------------</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>

**BMCHIST table considerations for COPY PLUS**

COPY PLUS uses the BMCHIST table to record completed COPY and COPY IMAGECOPY command executions.

HISTRETN is available as a COPY PLUS installation option or as an option on the OPTIONS command. HISTRETN tells COPY PLUS the number of days to keep entries in the BMCHIST table.

**WARNING**

If you want to use BMCHIST, allocate adequate space for the table. COPY PLUS makes an entry in the table for every copied space. If you are copying a large number of partitions, you might need to increase the size of the BMCHIST table space from the standard size that was allocated during installation.

**BMCHIST table considerations for RECOVER PLUS**

For each execution of AFRMAIN, RECOVER PLUS writes a single row to the BMCHIST table.

DBNAME, SPNAME, and OBJNAME columns will always be blank.

RECOVER PLUS accumulates elapsed time for each of the following phases using the RECOVER PLUS phase shown:

- PHASE_1: LOGSORT
- PHASE_2: MERGE (includes RESTORE phase)
- PHASE_3: SNAP
- PHASE_4: REBUILD (includes UNLOAD phase)
- PHASE_5: DB2UTIL (the time spent in DSNUTILB)

The elapsed time for each of the phases is a sum for all objects. The utility elapsed time, ELAPSED, is the duration from the start of the utility to until it finishes. Because RECOVER PLUS multitasks, the sum of the phases might be greater than the total elapsed time of the utility. The elapsed time columns have a limit of 24 hours.

Maintaining the BMCHIST table

When a utility completes successfully, it inserts a row into the BMCHIST table. You can control expansion of this table by deleting old rows. If you use REORG PLUS, you can also control inserts into the BMCHIST table.

To delete old rows from the BMCHIST table

1. To delete selected rows from the BMCHIST table based on the date that the utility completed, use the following sample SQL statement:

   ```sql
   DELETE
   FROM creatorName.CMN_BMCHIST
   WHERE DATE < 'yyyy-mm-dd';
   ```

To control inserts into the BMCHIST table (REORG PLUS only)

1. Use the TERMEXIT option to specify a user exit that controls inserts into the BMCHIST table.

BMCLGRNX table

The BMCLGRNX table contains log ranges that show when a table space was open for updates.

Table 35 on page 631 describes the contents of the BMCLGRNX table.

**Table 35: Contents of the 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 (mmdyy)</td>
</tr>
<tr>
<td>Column name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>LGRUCTM</td>
<td>CHAR(8)</td>
<td>Modification time (hhmmsssth)</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>

**Note**

RECOVERY MANAGER uses the BMCLGRNX table only for DB2 Versions 9 and 10. RECOVERY MANAGER uses the SYSIBM.SYSLGRNX table for DB2 versions greater than Version 10.

**BMCSYNC table**

The BMCSYNC table contains information about the status of the objects that the currently executing utilities are accessing.

Table 36 on page 632 describes the contents of the BMCSYNC table. 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 36: Contents of the BMCSYNC 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 (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>Column name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| NAME1      | CHAR(8)     | Database name or creator name  
(DASD MANAGER PLUS) This value is the database name.  
(CHECK PLUS, LOADPLUS, REORG PLUS, and UNLOAD PLUS) If the value for NAME1 would exceed 8 bytes or the value for NAME2 would exceed 18 bytes, NAME1 contains the DBID for the object. |
| NAME2      | CHAR(18)    | Space, table, or index name  
(DASD MANAGER PLUS) The BMCSTATS utility always inserts the space name (limited to a maximum of 8 characters).  
(CHECK PLUS, LOADPLUS, REORG PLUS, and UNLOAD PLUS) If the value for NAME1 would exceed 8 bytes or the value for NAME2 would exceed 18 bytes, NAME2 contains the table OBID or index ISOBID of the object in hexadecimal format. |
| KIND       | CHAR(2)     | Type of object:  
■ IP (index partition)  
■ IX (index)  
■ TB (table)  
■ TP (table space partition)  
■ TS (table space)  
■ DD, DW, D1, D2 (dynamic work file allocation)  
■ CI (copy information)  
■ RD (restart data set block) |
| PARTITION  | SMALLINT    | Physical partition number:  
■ Null or 0 for a single data set nonpartitioned space  
■ Data set number for a multi-data-set, nonpartitioned space  
■ Partition number for a partitioned space  
(CHECK PLUS, COPY PLUS, DASD MANAGER PLUS, LOADPLUS, REORG PLUS, and UNLOAD PLUS) The value is null or 0 for any nonpartitioned space. |
<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMCID</td>
<td>SMALLINT</td>
<td>Internal identifier of the object. DASD MANAGER PLUS does not use this column.</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>
<tr>
<td>SHRLEVEL</td>
<td>CHAR(1)</td>
<td>Degree to which utilities can share this object:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Blank means that no status is requested, and any other utility can obtain any status.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- S allows sharing among any number of SHRLEVEL S utilities.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- X indicates that exclusive control is required. No other utility can run with SHRLEVEL X.</td>
</tr>
<tr>
<td>STATUS</td>
<td>CHAR(1)</td>
<td>Status of the utility or object:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Blank (indicates no processing has been done)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- C (for CHECK PLUS, indicates checked)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- L (for LOADPLUS, indicates loaded)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- U (for UNLOAD PLUS, indicates unloaded)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- R (for REORG PLUS, indicates reloaded)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DASD MANAGER PLUS does not use this column.</td>
</tr>
</tbody>
</table>
BMCSYNC table considerations

This topic contains important information that you need to know about the BMCSYNC table:

- By default, DASD MANAGER PLUS uses the BMCSYNC table to synchronize access to DB2 spaces. However, if you want to turn this feature off, you may do so by specifying No for the BMCSYNC installation option. If you specify No for this option, DASD MANAGER PLUS does not use the BMCSYNC table and the product bypasses BMCUTIL table access, UTILID enqueue logic, and object name enqueue logic used for BMC utility concurrency control. Turning this feature off can lead to VSAM data set access failures in BMCSTATS or other utilities due to utility conflicts that are no longer detected.
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 exists:

— 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

— You are loading a partition-by-growth table space.
  Estimate this allocation based on the following factors:
  — Number of utilities that you are running concurrently
  — Value of MAXPARTITIONS
  — Number of files that you are allocating dynamically

— You are loading or unloading XML data and the XML table space is partition-by-growth.
  Estimate this allocation based on the following factors:
  — Number of utilities that you are executing concurrently
  — Number of XML columns that you are loading or unloading
  — Value of MAXPARTITIONS (a minimum of 256 partitions in this case)
  — Number of files that you are allocating dynamically

— You are loading or unloading LOB data.
  Estimate this allocation based on the following factors:
  — Number of utilities that you are executing concurrently
  — Number of LOB columns that you are loading or unloading
  — Number of partitions in the base table space
  — Number of files that you are allocating dynamically
Maintaining the BMCSYNC table

When a utility abends, rows might remain in the BMCSYNC table. On rare occasions, you might need to take action to control expansion of the BMCSYNC table.

**To control expansion of the BMCSYNC table**

1. Use one of the following methods to delete rows in the BMCSYNC table:
   - Use the TERM restart parameter on the EXEC statement to delete rows from both the BMCUTIL and BMCSYNC tables. Do not delete any rows for instances of utilities that are awaiting restart.
   - Delete invalid rows from 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.

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.

**To clean up RECOVER UNLOADKEYS entries**

1. Use one of the following methods to remove the invalid BMCSYNC rows:
   - Run a RECOVER BUILDINDEX job.
   - Run a job that uses the following statement for the table space and each index:
     ```sql
     DELETE FROM creatorName.CMN_BMCSYNC
     WHERE UTILID=''
     AND NAME1='databaseName'
     AND NAME2='spaceName'
     AND UTILNAME='RECOVER';
     ```
Shared access levels of BMC utilities

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.

**Note**

All BMC utility products use the BMCUTIL table to control the use of utility IDs, which identify executions of BMC utilities. Each BMC utility product must have a unique ID for restart purposes. This unique ID is stored in the BMCUTIL table. For more information about this table, see “BMCUTIL table” on page 642.

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 the following table. For additional serialization and concurrency issues for each utility, see that utility’s reference manual.

The "Access level" column in the following table refers to the value of the SHRLEVEL column name in the BMCSYNC table (“BMCSYNC table” on page 632). 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 37: Shared access levels of BMC utilities**

<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>Product</td>
<td>Access level</td>
<td>Additional information</td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| DASD MANAGER PLUS (BMCSTATS) | S            | ■ 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. |
| LOADPLUS                     | X            | 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. |
| RECOVER PLUS                 | X, S, or blank| RECOVER PLUS registers an object with shared access (S) under the following conditions:                                                                 |
■ 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.  
■ A table space partition is registered with shared access if the keys for that partition are unloaded with a RECOVER UNLOADKEYS operation.  
RECOVER PLUS registers an object with no access status (blank) if you specify the following commands or options:  
■ The ACCUM command  
■ OUTCOPY ONLY  
■ INDEP OUTSPACE  
RECOVER PLUS registers the object with exclusive access (X) in all other cases. |
| RECOVERY MANAGER             | S            | None                                                                                                                                                  |
| REORG PLUS                   | X            | 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. |
| UNLOAD PLUS                  | S            | None                                                                                                                                                  |
**WARNING**
Do not run an IBM utility, command, or SQL statement that attempts to manipulate the structure, data, or status of an object that a BMC utility is currently processing. For example, commands and SQL statements such as -STOP, -START, EXCHANGE, and ALTER will produce unpredictable results.

---

**BMCTRANS table**

The BMCTRANS table contains information that RECOVERY MANAGER and Log Master use for transaction recovery.

Table 38 on page 640 describes the contents of the BMCTRANS table. The table contains one row for each execution of Log Master (that is, one row for each log scan performed).

**Table 38: Contents of the BMCTRANS table**

<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>
<tr>
<td>PITRBA</td>
<td>CHAR(6) NOT NULL FORBIT 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>STATE</td>
<td>SMALLINT NOT NULL</td>
<td>Level of recovery analysis performed:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 0 (only UNDO analysis has been performed)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 1 through 9999 (UNDO and PIT analysis have been performed)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Greater than 10000 (UNDO, PIT, and REDO analysis have been performed)</td>
</tr>
<tr>
<td>PITTIME</td>
<td>TIMESTAMP NOT NULL WITH DEFAULT</td>
<td>Timestamp for the PIT RBA</td>
</tr>
<tr>
<td>Column Name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</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>Column Name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------</td>
<td>----------------------------------------------------------------------------</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>
<tr>
<td>ENDTIME</td>
<td>TIMESTAMP NOT NULL WITH</td>
<td>Transaction end time</td>
</tr>
<tr>
<td></td>
<td>DEFAULT</td>
<td></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>

**BMCUTIL table**

The BMCUTIL table contains information about utilities that are currently running or started.

 таблица contains information about utilities that are currently running or started.

Table 39 on page 643 describes the contents of the BMCUTIL table. 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 39: Contents of the BMCUTIL 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>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><em>(DASD MANAGER PLUS)</em> The value for this column is always 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>
<tr>
<td>PHASE</td>
<td>CHAR(8)</td>
<td>Current phase of the utility</td>
</tr>
<tr>
<td>USERID</td>
<td>CHAR(8)</td>
<td>User ID executing the utility</td>
</tr>
<tr>
<td>SSID</td>
<td>CHAR(4)</td>
<td>DB2 subsystem where the utility is running</td>
</tr>
<tr>
<td>Column name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RESTART</td>
<td>CHAR(1)</td>
<td>Restart option:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ N (not restart)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ P (RESTART(PHASE))</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Y (RESTART)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DASD MANAGER PLUS does not use this column.</td>
</tr>
<tr>
<td>NOTEID</td>
<td>CHAR(8)</td>
<td>TSO user ID to be notified</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DASD MANAGER PLUS does not use this column.</td>
</tr>
<tr>
<td>DBNAME</td>
<td>CHAR(8)</td>
<td><em>(RECOVER PLUS and REORG PLUS)</em> Name of the database containing the table or index space for which the last checkpoint was taken</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value can be blank.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The other utilities do not use this column.</td>
</tr>
<tr>
<td>SPNAME</td>
<td>CHAR(8)</td>
<td><em>(RECOVER PLUS and REORG PLUS)</em> Name of the table or index space for which the last checkpoint was taken</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value can be blank.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The other utilities do not use this column.</td>
</tr>
<tr>
<td>SPSTATUS</td>
<td>CHAR(5)</td>
<td><em>(REORG PLUS)</em> Space status before the utility stopped</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The other utilities do not use this column.</td>
</tr>
<tr>
<td>COMMANDNO</td>
<td>SMALLINT</td>
<td>Not used (always 0)</td>
</tr>
<tr>
<td>COMMAND</td>
<td>VARCHAR(256)</td>
<td>First 256 characters of the utility command text</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVER PLUS, DASD MANAGER PLUS, and COPY PLUS do not use this column.</td>
</tr>
<tr>
<td>STATE</td>
<td>LONG VARCHAR</td>
<td>Utility state and sync information</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DASD MANAGER PLUS does not use this column.</td>
</tr>
<tr>
<td>START_TIMESTAMP</td>
<td>TIMESTAMP</td>
<td>Starting timestamp of the utility</td>
</tr>
</tbody>
</table>
Maintaining the BMCUTIL table

When a utility abends, rows might remain in the BMCUTIL table.

On rare occasions, you might need to take action to control expansion of the BMCUTIL table.

To control expansion of the BMCUTIL table

1 Use one of the following methods to delete rows from the BMCUTIL table:

- Use the TERM restart parameter on the EXEC statement to delete rows from both 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

The BMC utilities use the BMCXCOPY table to track registered copies.

Table 40 on page 646 describes the contents of the BMCXCOPY table, which contains information about 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 that are not registered as Flash Copies in SYSCOPY 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

- Online consistent copies

- 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 40: Contents of the BMCXCOPY 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</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>
</tbody>
</table>
### BMCXCOPY table

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
</table>
| ICTYPE      | CHAR(1)   | Operation type:  
  - F (COPY FULL YES; for COPY PLUS, online consistent copies)  
  - I (COPY FULL NO)  
  - W (REORG LOG NO)  
  - B (REBUILD INDEX)  
  - P (POINT-IN-TIME RECOVERY)  
  - C (for COPY PLUS version 7.3 and earlier, online consistent copies)  
  - s (used by COPY PLUS to track system pages)  
  - m (indicates that the table space was exported by the COPY PLUS EXPORT command or migrated by the RECOVER PLUS IMPORT command) |
| ICDATE      | CHAR(6)   | Date of the entry (yymmdd) |
| START_RBA   | VARCHAR(10)| The relative byte location of a point in the DB2 recovery log  
  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 |
<p>| FILESEQNO   | INTEGER   | Tape file sequence number of the copy |
| DEVTYPE     | CHAR(8)   | Type of device on which the copy resides |</p>
<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
</table>
| IBMREQD      | CHAR(1)        | Whether the row came from the basic machine-readable material (MRM) tape:  
|              |                | ■ N (NO)  
|              |                | ■ Y (YES)  |
| DSNNAME      | CHAR(44)       | Name of the data set  
|              |                | If STYPE V, DSNNAME is the name of the VSAM data component.  |
| ICTIME       | CHAR(6)        | Time at which this row was inserted (hhmmss)  
|              |                | The insertion takes place after the completion of the operation that the row represents.  |
| SHRLEVEL     | CHAR(1)        | SHRLEVEL parameter on COPY if ICTYPE F:  
|              |                | ■ C (change)  
|              |                | ■ R (reference)  |
| DSVOLSER     | VARCHAR(1784)  | Volume serial numbers of the data set  
|              |                | Commas separate items in a list of 6-byte numbers. This column is blank if the data set is cataloged.  |
| TIMESTAMP    | TIMESTAMP      | Date and time when the row was inserted  
|              |                | This column contains 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.  |
| ICBACKUP     | CHAR(2)        | Type of image copy contained in the data set:  
|              |                | ■ LB (data set contains local backup data)  
|              |                | ■ RP (data set contains recovery system main data)  
|              |                | ■ RB (data set contains recovery system backup data)  
<p>|              |                | ■ Blank (data set contains local system main data or is not one of multiple copies)  |</p>
<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>PIT_RBA</td>
<td>VARCHAR(10)</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(53)</td>
<td>Number of pages written to the copy data set</td>
</tr>
<tr>
<td>NPAGESF</td>
<td>FLOAT(53)</td>
<td>High-used RBA divided by the page size</td>
</tr>
<tr>
<td>CPAGESF</td>
<td>FLOAT(53)</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>Column name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</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. 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></td>
<td></td>
<td>RBA/LRSN format for the space or partition:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ B (basic 6-byte format)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ E (extended 10-byte format)</td>
</tr>
<tr>
<td>INSTANCE</td>
<td>SMALLINT</td>
<td>Instance number of the current base objects (table and index). The default value is 1.</td>
</tr>
<tr>
<td>RELCREATED</td>
<td>CHAR(1)</td>
<td>DB2 release that created the object. If the release is earlier than Version 9, the value is blank.</td>
</tr>
<tr>
<td>COPY_TYPE</td>
<td>CHAR(1)</td>
<td>Type of copy:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ C (cabinet copy)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ O (online consistent copy)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X (export copy)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ I (import copy)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Blank (default value)</td>
</tr>
</tbody>
</table>
### BMCXCOPY table

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOTE_VALUE</td>
<td>CHAR(4)</td>
<td>Encoded value that quickly locates data for a specific space in a cabinet copy. The default value is blank.</td>
</tr>
<tr>
<td>NOTE_TYPE</td>
<td>CHAR(1)</td>
<td>Type of NOTE (issued by COPY PLUS):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ A (ABS - tape)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ R (REL - disk)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ F (frame)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Blank (default value)</td>
</tr>
<tr>
<td>OCC_COPY_RBA</td>
<td>VARCHAR(10)</td>
<td>Original START_RBA of an online consistent copy.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is blank.</td>
</tr>
<tr>
<td>OCC_LOCKRULE</td>
<td>CHAR(1)</td>
<td>Locking rule for a table space (not used for indexes):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ A (for page level)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ R (for row level)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Blank (default value)</td>
</tr>
<tr>
<td>OCC_SPACE_ALTERED</td>
<td>CHAR(1)</td>
<td>Whether the space was altered:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Y (altered)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ N (not altered)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Blank (default value)</td>
</tr>
<tr>
<td>CAB_BLOCKS</td>
<td>INTEGER</td>
<td>Total number of frames written for a cabinet copy</td>
</tr>
<tr>
<td>EXPSSID</td>
<td>VARCHAR(8)</td>
<td>Source location SSID of the migration file (valid with COPY_TYPE = I)</td>
</tr>
<tr>
<td>EXPSLRSN</td>
<td>VARCHAR(10)</td>
<td>Indicates the SYNC AUTO point on the source (valid with COPY_TYPE = I and COPY_TYPE = X)</td>
</tr>
<tr>
<td>EXPTLRSN</td>
<td>VARCHAR(10)</td>
<td>Indicates the SYNC AUTO point on the target (valid with COPY_TYPE = I)</td>
</tr>
</tbody>
</table>
Maintaining the BMCXCOPY table

Periodically, you should review BMCXCOPY and delete old rows to control its expansion.

To control expansion of the BMCXCOPY table

1 To delete all rows from the BMCXCOPY table that are older than 30 days, run an SQL DELETE statement, using the following statement as an example:

```sql
DELETE
FROM creatorName.CMN_BMCXCOPY
WHERE DAYS(CURRENT TIMESTAMP) - DAYS(TIMESTAMP) > 30;
```
BMC Common DB2 repository tables

The BMC common DB2 repository is made up of several DB2 tables.

Naming conventions

The BMC common DB2 repository tables follow a naming convention. The following table provides the synonyms and local table names.

Note
The local table names might be different at your site, based on options selected during product installation.

<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_GRPOPTS</td>
<td>BMCUTIL.CMN_OS_OPTS</td>
</tr>
<tr>
<td>BMCSCC_PRODREG</td>
<td>BMCUTIL.CMN_OS_PREG</td>
</tr>
<tr>
<td>BMCSCC_GROUPOUTH</td>
<td>BMCUTIL.CMN_OS_GAUTH</td>
</tr>
</tbody>
</table>

OBJSETS table

The following table 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>
<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>TIMESTAM 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>TIMESTAM 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></td>
<td></td>
<td>■ ACP (COPY PLUS)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ AFR (RECOVER PLUS)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ARM (RECOVERY MANAGER)</td>
</tr>
<tr>
<td>TYPE</td>
<td>CHAR(2) NOT NULL</td>
<td>Product group type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ BG - full subsystem</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ BA - application</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ RP - repository plan group</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>
<tr>
<td>OSNAME_DELIMITED</td>
<td>CHAR(1) NOT NULL WITH DEFAULT 'N'</td>
<td>For use with delimited names</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Will be 'Y' if the related column is a delimited name (entered with</td>
</tr>
<tr>
<td></td>
<td></td>
<td>double quotes around it) when entered</td>
</tr>
</tbody>
</table>
## OBJSET_DEF table

The following table 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>
<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>SEQNO</td>
<td>SMALLINT NOT NULL</td>
<td>Sequence number of definition</td>
</tr>
<tr>
<td>INCEXC_IND</td>
<td>CHAR(1) NOT NULL</td>
<td>Include or exclude indicator (+, -)</td>
</tr>
<tr>
<td>PATTERN_TYPE</td>
<td>CHAR(2) NOT NULL</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 (storgroup 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) NOT NULL</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) NOT NULL</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>Column name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>INC_LOBS</td>
<td>CHAR(1) NOT NULL</td>
<td>Include LOB objects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ <strong>Y</strong> <em>(Yes)</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ <strong>N</strong> <em>(No)</em></td>
</tr>
<tr>
<td>INC_XML</td>
<td>CHAR(1) NOT NULL</td>
<td>Include XML objects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ <strong>Y</strong> <em>(Yes)</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ <strong>N</strong> <em>(No)</em></td>
</tr>
<tr>
<td>INC_CLONES</td>
<td>CHAR(1) NOT NULL</td>
<td>Include clones only</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ <strong>Y</strong> <em>(Yes)</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ <strong>N</strong> <em>(No)</em></td>
</tr>
<tr>
<td>BY_PART</td>
<td>CHAR(1) NOT NULL</td>
<td>Expand objects by partition</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ <strong>Y</strong> <em>(Yes)</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ <strong>N</strong> <em>(No)</em></td>
</tr>
<tr>
<td>PART_BEG</td>
<td>SMALLINT NOT NULL</td>
<td>Beginning partition number (0-4096)</td>
</tr>
<tr>
<td>PART_END</td>
<td>SMALLINT NOT NULL</td>
<td>Ending partition number (0-4096)</td>
</tr>
<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>
</tbody>
</table>
## OBJSET_SQL table

The following table describes the contents of the OBJSET_SQL table. This table contains one row for each object set specification in dynamic SQL (type SQ).

<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>
**GRPOPTS table**

The following table 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.

<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>OPTION_TYPE</td>
<td>CHAR(10) NOT NULL</td>
<td>Option type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• backup—ARMOPTBKUP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• recover—ARMOPTRCVR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>These are the option types currently used by RECOVERY MANAGER. The option type is defined by the product, so this list is product-dependent.</td>
</tr>
<tr>
<td>OPTION</td>
<td>VARCHAR(200) NOT NULL</td>
<td>Option name</td>
</tr>
<tr>
<td>OPT_VALUE</td>
<td>VARCHAR(200) NOT NULL</td>
<td>Value for named option</td>
</tr>
</tbody>
</table>

**PRODREG table**

The following table describes the contents of the PRODREG table. There should be one entry for each product and version that is registered.

<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>

**GROUPAUTH table**

The following table 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>
<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 WITH DEFAULT</td>
<td>Timestamp of when authorization was granted</td>
</tr>
</tbody>
</table>
RECOVER PLUS syntax diagrams

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

RECOVER PLUS options are shown in the following table, alphabetized by RECOVER PLUS command, and within the command by option name.
## Table 41: RECOVER PLUS command options

<table>
<thead>
<tr>
<th>Command name</th>
<th>Reference</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMPORT</td>
<td>“IMPORT” on page 297</td>
<td>ANALYZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CHANGE ... FROM...TO</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXCLUDE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for OBJECTSET specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for USING specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP OUTSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOCOPYPEND</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSCRATCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPYDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REBUILD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REUSE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RICHK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SYNC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UPDATE VERSIONS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>USING</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>LOGSCAN</td>
<td>“LOGSCAN” on page 261</td>
<td>AUX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BACKOUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMLOGPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMRBA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for single index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for multiple index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXES (or INDEX)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for single index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for multiple index specification</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>LOGSCAN, continued</td>
<td>“LOGSCAN” on page 261</td>
<td>LOGAPPLY ONLY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGONLY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TOLOGPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTCOMMONQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• X’logPoint</td>
</tr>
<tr>
<td>TORBA</td>
<td></td>
<td>• LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTCOMMONQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• X’logPoint</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
<td>---------</td>
</tr>
<tr>
<td>MIGRATE</td>
<td>“MIGRATE” on page 277</td>
<td>ANALYZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CHANGE ... FROM...TO</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXCLUDE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for OBJECTSET specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for USING specification</td>
</tr>
<tr>
<td></td>
<td>FROMLOGPOINT</td>
<td>▪ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td>FROMRBA</td>
<td>▪ X'logPoint'</td>
</tr>
<tr>
<td></td>
<td>FROMRBA</td>
<td>▪ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td>FROMRBA</td>
<td>▪ LASTCOPY</td>
</tr>
<tr>
<td></td>
<td>FROMRBA</td>
<td>▪ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td>FROMRBA</td>
<td>▪ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td>FROMRBA</td>
<td>▪ X'logPoint'</td>
</tr>
<tr>
<td></td>
<td>FROMRBA</td>
<td>▪ X'logPoint'</td>
</tr>
<tr>
<td></td>
<td>INDEP OUTSPACE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LOGAPPLY ONLY</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LOGONLY</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LOGSORT</td>
<td></td>
</tr>
</tbody>
</table>

Appendix E  RECOVER PLUS syntax diagrams 665
<table>
<thead>
<tr>
<th>Command name</th>
<th>Reference</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIGRATE, continued</td>
<td>“MIGRATE” on page 277</td>
<td>MODEL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NUMREC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ABS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ AVGRECSZ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ CALC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ EST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ NOEST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TOCOPY LASTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TOLOCATION</td>
</tr>
<tr>
<td>MIGRATE, continued</td>
<td>“MIGRATE” on page 277</td>
<td>TOLOGPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOMMONQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE (relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LOGMARK logMarkName (logMarkGeneration)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td>TORBA</td>
<td></td>
<td>TOLOGPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOMMONQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE (relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LOGMARK logMarkName (logMarkGeneration)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td>USING</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>“OPTIONS” on page 113</td>
<td>ANALYZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTOSIZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BACKOUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CHECKINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DATAMVR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EARLYCAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EARLYRECALL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXLOG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IXRECP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>KEYSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>KSORTSHARE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGPOINT X'logPoint' or LOGPOINT LOGMARK logMarkName (logMarkGeneration)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOGSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXDRIVES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXKSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXLOGS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXLSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOEARLYCAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOEARLYRECALL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSYSLGRNG</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>OPTIONS, continued</td>
<td>“OPTIONS” on page 113</td>
<td>NUMREC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ABS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ AVGRECSZ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ CALC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ EST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ NOEST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ON ERROR ANY CONTINUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ASCODED</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ BYPART</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESINV</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RESOURCE SELECTION</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ACCUMS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ COPIES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LOGS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RICHK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SIMULATE YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SMCORE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SNAP</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>OPTIONS, continued</td>
<td>“OPTIONS” on page 113</td>
<td>SORTDEVT&lt;br&gt;■ for KEYSORT&lt;br&gt;■ for LOGSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDIAG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDYN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM&lt;br&gt;■ for KEYSORT&lt;br&gt;■ for LOGSORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STOGROUP... USEORDER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TIMESTAMP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRTCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>URIDDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>USEACCUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>USEHDROBIDS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WTOR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XBMID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZIIP</td>
</tr>
</tbody>
</table>
### Alphabetical listing of RECOVER PLUS options

<table>
<thead>
<tr>
<th>Command name</th>
<th>Reference</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>“OUTPUT” on page 151</td>
<td>CATLG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DATACLAS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNAMES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EATTR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXPDT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MAXPRIM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MGMTCLAS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODELDCB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RETPD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STACK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STORCLAS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRTCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNIT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNITCNT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VOLUMES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VOLUMECNT</td>
</tr>
</tbody>
</table>

---

670  *RECOVER PLUS for DB2 Reference Manual*
<table>
<thead>
<tr>
<th>Command name</th>
<th>Reference</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>REBUILD INDEX</td>
<td>“REBUILD INDEX” on page 223</td>
<td>ANALYZE</td>
</tr>
<tr>
<td>SIMRBLD INDEX</td>
<td></td>
<td>AUX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP INTABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEP OUTSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
</tr>
<tr>
<td></td>
<td>■ for INDEP INTABLESPACE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ for INDEP OUTSPACE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSCRATCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NUMREC</td>
</tr>
<tr>
<td></td>
<td>■ ABS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ CALC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ EST</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ NOEST</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOWORKDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PART</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REUSE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WORKDDN</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
<td>------------------</td>
</tr>
<tr>
<td>RECOVER</td>
<td>“RECOVER BUILDINDEX” on page 254</td>
<td>ANALYZE</td>
</tr>
<tr>
<td>BUILDINDEX</td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td>SIMRCVR</td>
<td></td>
<td>INDEP OUTSPACE</td>
</tr>
<tr>
<td>BUILDINDEX</td>
<td></td>
<td>MODEL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOSCRATCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REDefine</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REUSE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SKEYDNN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>RECOVER INDEX</td>
<td>RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE” on page 161</td>
<td>ANALYZE</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>AUX</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>BACKOUT</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>CLONE</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>DBID</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>DEFINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DROPRECOVERY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNAME</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSNUM</td>
</tr>
<tr>
<td></td>
<td>■ for INDEX</td>
<td>DSSIZE</td>
</tr>
<tr>
<td></td>
<td>■ for INDXSPCACE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ for TABLESPACE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ENCRYPTED</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMLOGPOINT</td>
</tr>
<tr>
<td></td>
<td>■ LASTARCHQ</td>
<td>• LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td>■ LASTCOPY</td>
<td>• LASTCOPY</td>
</tr>
<tr>
<td></td>
<td>■ LASTQUIESCE</td>
<td>• LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td>■ LASTSHUTDOWN</td>
<td>• LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td>■ X’logPoint</td>
<td>• X’logPoint</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FROMRBA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• X’logPoint</td>
</tr>
</tbody>
</table>

Alphabetical listing of RECOVER PLUS options

Appendix E  RECOVER PLUS syntax diagrams 673
### Alphabetical listing of RECOVER PLUS options

<table>
<thead>
<tr>
<th>Command name</th>
<th>Reference</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX</td>
<td>“RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE” on page 161</td>
<td>INCOPY FULL</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>INCR</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>INDEP OUTSPACE</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>INDEVT</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>INDEX</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>■ for single index specification</td>
</tr>
<tr>
<td>(continued)</td>
<td></td>
<td>■ for multiple index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEXES (or INDEX)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for single index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for multiple index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INLINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INLOG LOGPOINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INLOG RBA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSEQNO</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INVOLUME</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LASTCOPY (relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOCALSITE</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>RECOVER INDEX</td>
<td></td>
<td>LOGAPPLY ONLY</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>LOGONLY</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>LOGPOINT</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>LOGSORT</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>MODEL</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>NOCOPYPEND</td>
</tr>
<tr>
<td>(continued)</td>
<td></td>
<td>NOSCRATCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOWORKDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NUMREC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBIDXLAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJECTSET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPYDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTCOPYDSN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PIECESIZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RBA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYDDN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYDSN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVERYSITE</td>
</tr>
</tbody>
</table>

"RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE" on page 161
### Alphabetical listing of RECOVER PLUS options

<table>
<thead>
<tr>
<th>Command name</th>
<th>Reference</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER INDEX</td>
<td>“RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE” on page 161</td>
<td>REDEFINE</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>REGISTER</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>RESET</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>RESETRTS</td>
</tr>
<tr>
<td>SIMRCVRL INDEXSPACE</td>
<td></td>
<td>REUSE</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>RICHK</td>
</tr>
<tr>
<td>(continued)</td>
<td></td>
<td>SHRLEVEL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SNAPSHOT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTDEVT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SORTNUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABLESPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for table space specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ for multiple index specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TOCOPY</td>
</tr>
<tr>
<td>Command name</td>
<td>Reference</td>
<td>Options</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>RECOVER INDEX</td>
<td>“RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE” on page 161</td>
<td>TOLOGPOINT</td>
</tr>
<tr>
<td>RECOVER INDEXSPACE</td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td>RECOVER TABLESPACE</td>
<td></td>
<td>■ LASTCOMMONQ</td>
</tr>
<tr>
<td>SIMRCVR INDEX</td>
<td></td>
<td>■ LASTQUIESCE</td>
</tr>
<tr>
<td>SIMRCVR INDEXSPACE</td>
<td></td>
<td>(relativeGenerationNumber)</td>
</tr>
<tr>
<td>SIMRCVR TABLESPACE</td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td>(continued)</td>
<td></td>
<td>■ LOGMARK logMarkName</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(logMarkGeneration)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td>TORBA</td>
<td></td>
<td>■ LASTARCHQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTCOMMONQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTQUIESCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(relativeGenerationNumber)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LASTSHUTDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LOGMARK logMarkName</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(logMarkGeneration)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ X’logPoint</td>
</tr>
<tr>
<td>TOSEQNO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOVOLUME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRANSFORM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPDATE VERSIONS</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The following syntax diagrams define the options that can be used with the OPTIONS command.
Figure 190: OPTIONS syntax diagram

OPTIONS

OPTION

EARLYRECALL NOEARLYRECALL

EARLYCAT NOEARLYCAT

LOGSORT work data set specification

NOSYSLGRNG

USEACCUM YES NO

CHECKINT (integer)

STOGROUP specification

SORT specification

MAXDRIVES integer

MAXLOGS integer

OUTCOPY

ASCODED BYPART

RICHK NO YES

AUX

NO

ALL

XML

LOB

USEHDROBIDS YES NO

HISTORY

1 The default for ANALYZE ONLY runs
Figure 191: OPTIONS syntax diagram, continued

1 Valid with R+/CHANGE ACCUM only
2 Requires a Recovery Management for DB2 solution password
Figure 192: OPTIONS syntax diagram, continued

OPTIONS command syntax diagram

KEYSORT

SORTDEVT deviceType

SORTNUM

ANALYZE

YES

NO

ONLY

SNAP

HW

VSAM

SIMULATE YES'

LOGPOINT X'logPoint'

WTOR

YES

NO

ON ERROR

CONTINUE integer

MAXKSORT integer

MAXLSORT integer

ON ERROR

CONTINUE integer

ANY

KSORTSHARE

YES

NO

AUTOSIZE

YES

NO

RECOVERYPOINT'

TIMESTAMP yyyy-mm-dd-h.mm.ss.ttttt

URIDDN (output)

LOGPOINT

X'logPoint'

LOGMARK logMarkName

(logMarkGeneration)

'Requires a Recovery Management for DBZ solution password
Figure 193: OPTIONS syntax--LOGSORT specification

**LOGSORT specification**

- LOGSORT
  - SORTDEVT `deviceType`
  - SORTNUM `integer`
  - NUMREC
    - CALC
    - NOEST
    - EST `integer`
    - ABS `integer`
      - AVGRECSZ `integer`

Figure 194: OPTIONS syntax--STOGROUP specification

**STOGROUP specification**

- STOGROUP `storageGroupName`
  - USEORDER
    - `volumeSerialNumber`

Figure 195: OPTIONS syntax--SORT specification

**SORT specification**

- SORTDIAG `NO`
  - YES
- SORTDYN `YES`
  - NO
- RESINV `(integerK)`
- SMCORE `(integerK, integerK)`
The following syntax diagrams define the options that can be used with the OUTPUT command.

**Figure 196: OUTPUT syntax**

```
OUTPUT  name  DSNAMES dataSetName
```

**Common options**

```
UNIT  SYSLDATA  name
CATLG  YES  NO
MODELDCB  dataSetName
VOLCNT 25
UNITCNT 0
DATACLAS  name
MGMTCLAS  name
STORCLAS  name
```

**Disk options**

```
SPACE  (primary,secondary)
CYL  TRK
VOLUMES  (volume1,volume2,...)
EATTR  NO  OPT  NONE
MAXPRIM  0
```

Appendix E  RECOVER PLUS syntax diagrams 683
The following syntax diagrams define the options that can be used with the RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE commands.
Figure 200: RECOVER TABLESPACE, RECOVER INDEX, and RECOVER INDEXSPACE syntax diagram

RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE

- RECOVER SIMRCVR
  - Table space specification
    - INDEPENDENT OUTSPACE specification
    - TORBA or TOLOGPOINT recovery specification
  - Single index specification
    - Independent OUTSPACE specification
    - TOCOPY specification
    - Non-registered copy or INLOG specification
  - Multiple index specification
    - OBJECTSET specification
      - Table space specification
        - Single index specification
          - OBJECTSET specification
        - OBJECTSET specification

- LOGSORT specification
  - LOGAPPLY ONLY
    - LOCALSITE RECOVERYSITE
  - CLONE
  - DEFINE
    - NO
    - YES

- ANALYZE
  - YES
  - NO
  - ONLY

- REDEFINE
  - YES
  - NOSCRATCH
  - NO

- UPDATE VERSIONS

1 Not valid with BACKOUT.
Figure 201: RECOVER syntax--Table space specification

Table space specification

- `TABLESPACE` followed by `tableSpaceName`
- `DSNDB04`, `databaseName`
- `OBJECTSET` specification
- `INDEXES INDEX NO YES`
- `INDEX` specification
- `DSNUM` with `ALL` or `begin : end`
- `AUX` with `NO`, `ALL`, `XML`, `LOB`
- `DROPRECOVERY` with `NO YES`
- `RICHK` with `NO YES`
- `OBIDXLAT` specification
- `TRANSFORM` specification
- `Non-registered copy or INLOG specification`
- `FROMRBA` or `FROMLOGPOINT` specification
- `OUTCOPY` specification

1 Not valid with `TRANSFORM`
2 Not valid with `BACKOUT`
Figure 202: RECOVER syntax--Single index specification

Single index specification

INDEX (indexName — )
  creatorID.
  OBJECTSET specification

INDEXSPACE (databaseName. indexSpaceName — )
  DSND04.

DSNUM ALL
  integer
  begin : end

NOWORKDDN

OBIDXLAT specification

TRANSFORM

Non-registered copy or INLOG specification

FROMRBA or FROMLOGPOINT specification

OUTCOPY specification

¹ Valid with INDEXLOG AUTO.
Figure 203: RECOVER syntax--Multiple index specification

Multiple index specification

- **OBJECTSET** specification
  - `INDEX` (creatorID, `indexName`)
  - `INDEXSPACE` (`databaseName`, `indexSpaceName`)
    - `TABLESPACE` (`databaseName`, `tableSpaceName`)
      - `AUX`
      - `NO`
      - `ALL`
      - `XML`
      - `LOB`
      - `HISTORY`

- **NOWORKDDN**

- **OUTCOPY** specification
  - `INDEXLOG AUTO`

\(^1\) Allowed with multiple recoveries only if dynamic outcopy allocation (OUTPUT) syntax is used.

\(^2\) Valid with INDEXLOG AUTO.
**Figure 204: RECOVER syntax--OBJECTSET specification**

**OBJECTSET specification**

```
OBJECTSET creator.name

EXCLUDE

(dbname.ts qualifier.ix)

DSNUM ALL

integer

begin : end

OBIDXLAT

RESET
```

**Figure 205: RECOVER syntax--INDEPENDENT OUTSPACE specification**

**INDEPENDENT OUTSPACE specification**

```
INDEP OUTSPACE

MODEL vcat.BMCDBC.databaseName.objectName.0001.znnn

MODEL userNamedDataSet

^1 Not valid with BACKOUT
```
Figure 206: RECOVER syntax--TORBA or TOLOGPOINT specification

TORBA or TOLOGPOINT specification

TORBA
TOLOGPOINT
LASTQUIESCE
LASTCOMMONQ
LASTARCHQ
LASTSHUTDOWN
LOGMARK logMarkName
X\'logPoint\' (logMarkGeneration)
BACKOUT
YES

Figure 207: RECOVER syntax--TOCOPY specification

TOCOPY specification

TOCOPY
LASTCOPY
\textit{dateSetName} ^
TOVOLUME
CATALOG
volumeSerialNumber
TOSEQNO integer

\textit{NOT valid with OBJECTSET}
Figure 208: RECOVER syntax--LOGSORT specification

LOGSORT specification

- LOGSORT
  - SORTDEVT deviceType
  - SORTNUM integer
  - NUMREC
    - CALC
    - NOEST
    - EST integer
      - AVGRECSZ integer
    - ABS integer
      - AVGRECSZ integer
Figure 209: RECOVER syntax--OBIDXLAT specification

OBIDXLAT specification

1 Not valid with BACKOUT
Figure 210: RECOVER syntax--Non-registered copy or INLOG specification

Non-registered copy or INLOG specification

INCOPY — FULL
  INCOPY specification
  INCR ¹
  NOCOPYPEND
  INLOG RBA ¹·₁
  INLOG LOGPOINT ¹·₂
  X'logPoint' ¹

¹ Not valid with index specification
² Not valid with BACKOUT

Figure 211: RECOVER syntax--FROMRBA or FROMLOGPOINT specification

FROMRBA or FROMLOGPOINT specification

FROMRBA
FROMLOGPOINT
LASTQUIESCE
LASTCOPY
LASTARCHQ
LASTSHUTDOWN
X'logPoint'

Figure 212: RECOVER syntax--OUTCOPY specification

OUTCOPY specification

OUTCOPY
NO
YES
ONLY ¹
REGISTER
ALL
NONE
reset
1
ddname

¹ Not valid with BACKOUT or Instant Snapshot copies
Figure 213: RECOVER syntax--OUTCOPY specification, continued

OUTCOPY specification, continued

- OUTCOPYDDN specification
- RECOVERYDDN specification
- TAG tagName
- SPACE (pri,sec)
- CYL
- TRK

RECOVER TABLESPACE, RECOVER INDEX, RECOVER INDEXSPACE command syntax diagram
Figure 214: RECOVER syntax--INCOPY specification

INCOPY specification

- DSNNAME dataSetName
- MODEL dataSetName
  - DSIZE (integerG)¹
  - PIECESIZE (...)¹
  - integerK
  - integerM
  - integerG

- RBA
- LOGPOINT
- X'logPoint'
- SHRLEVEL
- REFERENCE
- CHANGE

- ENCRYPTED
- TIMESTAMP timestamp
- COMPRESSED ²

- INVOLUME
- CATALOG
  - volumeSerialNumber
  - INDEV deviceType
  - INSEQNO integer

¹ Valid only with TRANSFORM
² Not valid with table space specification

Appendix E  RECOVER PLUS syntax diagrams 695
Figure 215: RECOVER syntax--OUTCOPYDDN specification

\[
\text{OUTCOPYDDN specification}
\]

\[
\text{OUTCOPYDDN} \rightarrow \left( \begin{array}{c}
\text{BMCCPY}^{1,2} \\
\text{DDName1} \\
\text{DDNamePrefix1}^3 \\
\text{outputDescriptor1}^3 \\
, \end{array} \right) \left( \begin{array}{c}
\text{BMCCPZ}^{1,2} \\
\text{DDName2} \\
\text{DDNamePrefix2} \\
\text{outputDescriptor2}^3 \\
\end{array} \right)
\]

1. 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.
2. These values are the default values used by the OUTPUT command.
3. These values are used to specify the OUTPUT command, which will control the allocation of the copy.

Figure 216: RECOVER syntax--RECOVERYDDN specification

\[
\text{RECOVERYDDN specification}
\]

\[
\text{RECOVERYDDN} \rightarrow \left( \begin{array}{c}
\text{BMCRCY}^{1,2} \\
\text{DDName3} \\
\text{DDNamePrefix3} \\
\text{outputDescriptor3}^3 \\
\end{array} \right) \left( \begin{array}{c}
\text{BMCRCZ}^{1,2} \\
\text{DDName4} \\
\text{DDNamePrefix4} \\
\text{outputDescriptor4}^3 \\
\end{array} \right)
\]

\[
\text{RECOVERYDSN} \rightarrow \left( \begin{array}{c}
\text{dataSetName3} \\
\end{array} \right) \left( \begin{array}{c}
\text{dataSetName4} \\
\end{array} \right)
\]

1. 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.
2. These values are the default values used by the OUTPUT command.
3. These values are used to specify the OUTPUT command, which will control the allocation of the copy.
The following syntax diagrams define the options that can be used with the REBUILD INDEX command.

Figure 217: REBUILD INDEX syntax diagram

1 Is synonymous to REBUILD when INDEXLOG=NO
2 Is synonymous to SIMRBLD when INDEXLOG=NO
Figure 218: REBUILD INDEX syntax diagram, continued

Figure 219: REBUILD INDEX syntax diagram--OBJECTSET specification

OBJECTSET specification

OBJECTSET creator.name

EXCLUDE (dbname.ts qualifier.ix, DSNUM ALL integer begin : end)
The following syntax diagrams define the options that can be used with the RECOVER UNLOADKEYS command.
Figure 221: RECOVER UNLOADKEYS syntax diagram

```
RECOVER UNLOADKEYS (indexName PART\(^1\) integer begin : end)

TABLESPACE
    DSNDB04 databaseName.
    tableSpaceName
    PART\(^1\) integer begin : end

INDEP INTABLESPACE
    MODEL vcat.BMCDBC.databaseName.tableSpaceName.I0001.zrnn
    userModelDataSet

CLONE
    SKEYDDN SKEY DDName
    SORTDEVT deviceType
    SORTNUM integer

NUMREC
    CALC
    NOEST
    EST integer
    ABS integer

ANALYZE
    YES
    NO
    ONLY

\(^1\) You cannot specify PART in both places.
```
RECOVER BUILDINDEX command syntax diagram

The following syntax diagrams define the options that can be used with the RECOVER BUILDINDEX command.

Figure 222: RECOVER BUILDINDEX syntax diagram

LOGSCAN command syntax diagram

The following syntax diagrams define the options that can be used with the LOGSCAN command.
Figure 223: LOGSCAN syntax diagram

Figure 224: LOGSCAN syntax--Table space specification
Figure 225: LOGSCAN syntax---Single index specification

Single index specification

INDEX

OBJECTSET creator.name

INDEXSPACE

databaseName.

creatorID.

INDEX

indexName

INDEXSPACE

INDEX

databaseName.

DSNDB04

INDEXSPACE

INDEX

databaseName.

INDEXSPACE

FROMRBA

FROMLOGPOINT

LASTQUIESCE

LASTCOPY

LASTARCHQ

LASTSHUTDOWN

X'logPoint'
MIGRATE command syntax

The following syntax diagrams define the options that can be used with the MIGRATE command.

Figure 228: MIGRATE syntax

1 Requires a valid password as follows: Recovery Management or Database Administration
Figure 229: MIGRATE syntax--USING specification

**USING specification**

```
USING  dataSelName

CHANGE TABLE FROM creator1$tableName1 TO creator2$tableName2
CHANGE INDEX FROM creator1$indexName1 TO creator2$indexName2

INDEXES
RECOVER
NO

EXCLUDE
(dbname.ts qualifier.ix)

DSNUM
ALL
integer
begin : end

RESET```

Figure 230: MIGRATE syntax--FROMRBA or FROMLOGPOINT specification

**FROMRBA or FROMLOGPOINT specification**

```
FROMRBA
FROMLOGPOINT

LASTQUIESCE
LASTCOPY
LASTARCHQ
LASTSHUTDOWN
X'logPoint'
```
IMPORT command syntax

The following syntax diagrams define the options that can be used with the IMPORT command.
Figure 235: IMPORT syntax

IMPORT ^, USING specification

TABLESPACE |
<DSNDB04 Name>

tableSpace Name |

AUX |
NO |
ALL |
XML |
LOB |

HISTORY |
RICHK |
NO |
YES |

OBJECTSET specification |

INDEPENDENT OUTSPACE specification

OUTCOPY specification

CLONE

ANALYZE |
YES |
NO |
ONLY |

REDEFINE |
YES |
NO |

NOSCRATCH

REUSE

UPDATE VERSIONS

STATS |
NO |
YES |

DEFINE |
NO |
YES |

1 Requires a valid password as follows: Recovery Management or Database Administration
2 Not valid with BACKOUT
Figure 236: IMPORT syntax--USING specification

**USING specification**

```
USING dataSetName

CHANGE TABLE FROM creator1.tableName1 TO creator2.tableName2

CHANGE INDEX FROM creator1.indexName1 TO creator2.indexName2
```

```
EXCLUDE ( dbname.ts qualifier.ix )

DSNUM

ALL

integer

begin : end
```

```
SYNC AUTO REPLACE

INDEXES AUTO RECOVER REBUILD NO

RESET NOCOPYPEND
```

Figure 237: IMPORT syntax--OBJECTSET specification

**OBJECTSET specification**

```
OBJECTSET creator.name

EXCLUDE ( dbname.ts qualifier.ix )
```
Figure 238: IMPORT syntax--INDEPENDENT OUTSPACE specification

INDEPENDENT OUTSPACE specification

INDEP OUTSPACE

MODEL vcat.BMCDBC.databaseName.objectName.10001.znnn
MODEL userNamedDataSet

Figure 239: IMPORT syntax--OUTCOPY specification

OUTCOPY specification

OUTCOPY

NO YES

OUTCOPYDDN specification

RECOVERYDDN specification
Symbols

@x03DOPT 587
&ATTACH symbolic variable 153, 510
&DATE symbolic variable 153, 510
&DAY symbolic variable 153, 510
&DB symbolic variable 153, 510
&DSNUM symbolic variable 153, 510
&HOUR symbolic variable 153, 510
&ICTYPE symbolic variable 153, 510
&INST symbolic variable 153, 510
&JDATE symbolic variable 153, 510
&JDAY symbolic variable 153, 510
&JOBNAME symbolic variable 153, 510
&LDSNUM symbolic variable 153, 510
&LPART symbolic variable 153, 510
&MINUTE symbolic variable 153, 510
&MONTH symbolic variable 153, 510
&PART symbolic variable 153, 510
&PART5 variable 510
&SECOND symbolic variable 153, 510
&SEQ symbolic variable 153, 510
&SP symbolic variable 153, 510
&SSID symbolic variable 153, 510
&STEPNAM symbolic variable 153, 510
&TIME symbolic variable 153, 510
&TS symbolic variable 153, 510
&TYPE symbolic variable 153, 510
&UID symbolic variable 153, 510
&UNIQ (or &UQ) symbolic variable 153
&UNIQ (or &UQ) variable 510
&USERID symbolic variable 153, 510
&UTIL symbolic variable 153, 510
&VCAT symbolic variable 510
&YEAR symbolic variable 153, 510
$C30DOPT 587

A

above-the-bar storage 70
access, shared 632
ACCUM command 436
ACHKP status 60
added partitions 518
adding comments 90
adding objects for restart 342
ADRDBSU utility 552
Advanced Encryption Standard (AES) 521
advantages of LOGSORT strategy 474
advantages of sorting the log 473
AES (Advanced Encryption Standard) 521
AFR$OPTS installation options module 587
AFRDBG 326
AFRDUMP 326
AFRERR 326
AFR$DG data set 326, 509, 582
AFR MAIN execution module 342
AFROSUM 326
AFRPLAN 326
AFRPLAN data set 384
AFRPRT 326
AFRPn data set 326
AFRSTMT 326
AFRSTMT data set 384
AFRSUMRY 326
AFRSUMRY data set 384
AFRTIME 326
AFRTTRACE 326
allocating copy data sets 507
allocating output data sets 87, 153
allocation of EAV 157, 611
allocation of extended format sequential data set 157, 611
alphabetical listing of options 96, 661
ALTER for DB2 539, 540, 545, 549, 552, 557
AMSCAT option 594
ANALYZE ONLY option 384
ANALYZE option 295
   ANALYZE ONLY 54
      description 139, 201, 240, 253, 259
      description for IMPORT 311
      using to preview recovery resources and plans 54
ANSI Data Encryption Algorithm (DEA) 521
APF authorization 74, 587
archive log 467
ARCHIVE LOG MODE(QUIESCE) 564, 572, 582
archive quiesce points
   migrating from 286
   migrating to 290
   recovering from 210
   recovering to 193
   scanning from 265
   scanning to 274
ASCII data support 488
audit controls 36
authority
   APF 74
   DATAACCESS 72
   DBADM 72
   DBCTRL 72
   DISPLAYDB 72
   IMAGCOPY 72
   installation SYSADM 72
   RACF 73
   RECOVERDB 72
   STARTDB 72
   STOPDB 72
   SYSADM 72
   SYSCTRL 72
   system 73
   system DBADM 72
      to access DB2 subsystem 72
      to run RECOVER PLUS 71, 72
      to start the database 72
      to stop the database 72
authorizations
   mechanisms, description 71
   automatic fallback 479
   automatic sizing of output files 147
   AUTOSIZE option 147, 605
   AUX option 178, 185, 231, 265, 270, 304
      installation option description 608
   auxiliary table spaces 59
   AVGRECSZ option 47, 120

B

backing up BMC tables 623
BACKOUT AUTO option 51
BACKOUT AUTO option, description 131
BACKOUT option
   and hash spaces 483
   and indexes defined using an expression 483
   and LOBs 483
   and NOT LOGGED spaces 483
   and XML spaces 483
   description 131, 274
      example 414
      for TOLOGPOINT 193, 274
      for TORBA 193, 274
   installation option description 609
   recovery without image copies 48
      restrictions 131, 174, 193, 212, 274, 282, 483
   backout recovery 49
   backup copies, making 212
   benefits of RECOVER PLUS 35
   bind qualifier 606
   BINDQUALIFIER installation option 606
   BMC tables 653
   BMC utilities
      displaying status 625
      running concurrently 638
      terminating 625
   BMCDICT table
      considerations 627
      contents 626
      maintaining 627
   BMCHIST installation option, BMCHIST table 628
   BMCHIST table
      and the HISTORY installation option 606
      backing up 623
      contents 628
      COPY PLUS considerations 630
      maintenance 631
      querying 625
      RECOVER PLUS considerations 630
   BMCLGRNX table 631
   BMCSORT installation option 617
   BMCSYNC table
backing up 623
clean up after job termination 348
cleaning up RECOVER PLUS UNLOADKEYS 637
concurrency 79
considerations 635
contents 632
LOB data considerations 635
maintaining 637
running utilities concurrently 638
utilid deleted 320
XML data considerations 635
BMCTRANS table 640
BMCUUTIL table
backing up 623
clean up after job termination 348
concurrency 79
contents 642
maintaining 645
utilid deleted 320
BMXCOPY table
backing up 623
contents 645
maintaining 652
querying 625
registrations 57
role in index backups 61
build phase 366
BUILDINDEX option
description 256
rebuilding nonpartitioned indexes 53
building a nonpartitioned index 415
building RECOVER PLUS jobs 317

C
CA-ACF2 security product 71
CA-Top Secret security product 71
cabinet copies
registration 63, 64
strategies for use 500
overview 63
CATALOG MANAGER for DB2 539, 540, 545, 549, 552, 557
catalog, ICF 73
CATLG option 155
CHANGE ... FROM...TO option 283, 301
change accumulation file
described 503
dynamic sizing 147
grouping objects 504, 506
CHANGE MANAGER for DB2 539, 540, 545, 549, 552, 557, 558
changed limit keys 520
CHECKINT option
command option description 122
effects on restart 346
installation option description 593
checkpoints, specifying the time between 122
CHECKPT option
checkpoint override parameter 322
effects on restart 346
installation option description 593
CHECKPT(NO) 322
CHECKPT(PHASE) 322
cleaning up
after a job termination 348
after an UNLOADKEYS job 348
BMCUUTIL and BMCSYNC 348
using the TERM parameter 320
CLONE option 201, 235, 250, 258, 272, 295, 311, 529
cloned objects
CLONE option 529
data sets 529
limitations for RECOVER PLUS support 529
recovering 529
commands, alphabetical listing 96, 661
comments coded in statements 90
common DB2 repository tables 653
common repository
naming conventions 653
compatibility
BMC Software products 42
DB2 resources 42
compressed indexes 532
COMPRESSED option 217
compressing data sets, when stacking to tape 159
compression
BMCDICT table 626
concurrent operation
of UNLOADKEYS and BUILDINDEX 81
recovering multiple partitions 467
recovery of table space and indexes 472
with IBM DB2 utilities 78
with other BMC utilities for DB2 79
with other RECOVER PLUS executions 80
consistent point, obtaining 572, 582
controlling output 326
conventions, documentation 16
copies
migrating from the last 286
recovering from the last 210
scanning from the last 265
COPY command options
EXCLUDE 187, 229
copy data set ddnames 220–222
copy migration 535
COPY PLUS
long names in statements 91
utility integration 43
copy registration
BMCXCOPY table 645
copy registration, SHRLEVEL 581
copying cabinet copies 63
CPUMIPS installation option 613
creating
consistent database for query and review 558
consistent set of copies at a point in time 582
consistent set of copies at a point in time, OUTCOPY ONLY JCL 582
copies from log and prior copies 581, 582
DB2 objects 540, 545, 552, 558, 564
image copy without reading a space 584
migration images 572
creating copies from recovery resources 58
cumulative incremental image copies 82

D
data compression, for tape data sets 159
data definitions, obtaining 540, 545, 549, 552, 558, 564, 572
Data Encryption Standard (DES) 521
data migration 555
data sets

AFRDBG 326
AFRDUMP 326
AFRERR 326
AFRGDG 326
AFROSUM 326
AFRPLAN 326
AFRPRINT 326
AFRPRnnn 326
AFRSTMNT 326
AFRSUMRY 326
AFRTIME 326
AFRTRACE 326
compression, for tape data sets 159
dynamic allocation 87, 153, 325
for log 330
for output copies 220, 221
for REBUILD INDEX 333
for RECOVER BUILDINDEX 336
for RECOVER INDEX 330
for RECOVER INDEXSPACE 330
for RECOVER TABLESPACE 330
for RECOVER UNLOADKEYS 335
for sorts 336
input image copies 330
interim for index keys 53
L001WKnn 336, 340
log sort work 330
LxxxWKnn 336, 340
optional 326
output image copies 330
reallocating 203, 313
required 326
reusing 204, 242, 261, 314
S001WKnn 333, 336
SKEYDDN option 335, 336
STOGROUP-defined 203, 241, 260, 313
STOGROUP-defined, required authority 73
SxxxWKnn 333, 336
symbolic variables 190, 288, 308
SYSERR 326
SYSIN 326
SYSOUnnn 326, 333
SYSPICK 326
SYSPRINT 326
SYSSCAN 326
SYSUDUMP 326
tape stacking considerations 416
VCAT-defined 203, 241, 260, 313
VSAM 203, 241, 313
WORKDDN option 333
data sharing
   considerations 515
   group attach name 320
DATAACCESS authority 72
DATACLAS installation option 612
DATACLAS option 155
DATAMVR option 138, 606
date/time values
   Daylight Savings Time adjustments 150, 602
   TZRULE option 150, 602
Daylight Savings Time adjustments 602
DB2
   authorizations needed for RECOVER PLUS 71
   commands issued by RECOVER PLUS 75
   group attach name in JCL 320
   object status 323
   RACF ID 73
   subsystem ID (SSID) in JCL 320
   subsystem identifier (SSID) 320
DB2 objects, creating 540, 545, 552, 558, 564
DB2 REPAIR VERSIONS 528
DB2 versioning, steps to handle 528
DBADM authority 72
DBCTRL authority 72
DBID 540, 545, 558, 564, 572
DD name construction
   for nonpartitioned spaces 330
   for partitioned spaces 330
DD statements
   common to all RECOVER PLUS executions 326
   for REBUILD INDEX 333
   for RECOVER BUILDINDEX 336
   for RECOVER INDEX 330
   for RECOVER INDEXSPACE 330
   for RECOVER TABLESPACE 330
   for RECOVER UNLOADKEYS 335
   in JCL 325
input image copies 330
L001WKnn 336, 340
log data sets 330
LxxxWKnn 336, 340
output image copies 330
S001WKnn 333, 336
SKEYDDN option 335, 336
specifying work data sets for index keys 478
STEPLIB 325
SxxxWKnn 333, 336
SYSUT1 333
tape mounts, stacked tapes 330
WORKDDN option 333

DDNAME 415
DEA (Data Encryption Algorithm 521
default option See installation options 588
DEFINE option 201
   description of installation option 614
DES (Data Encryption Standard 521
DFDSS utility 552
dictionaries, compression
   BMCDICT table 626
directing recovery output 418
DISKIORATE installation option 613
DISPLAYDB authority 72
displaying job status 347
displaying status of BMC utilities 625
DocID 531
documentation
   summary of changes 18
documentation information 15
DOPT See installation options 587
drives, maximum 599
drop recovery
   dropped table from a simple table space 549
   dropped table in segmented table space 540
   overview 539
   using pack dump backup 552
   when image copies are available 540
dropped partitions 519
DROPRECOVERY option
   example of use 540, 545, 549, 552
   description 178
   example of use 540, 545, 549, 552
   restrictions 174
DSN1COPY with INCOPY 51
DSN1LOGP utility
   example use 540
   output example 549
   using to find LRSN 549
DSN1PRNT utility 540, 545
DSNAME option 153, 217
DSNHDECP for ssid 320
DSNUM option 182, 187, 229, 265, 269, 283, 301
DSNUTILB load module 74
DSSIZE option 217
DST adjustments 602
dynamic allocation
   DYNALLOC option, RETRY parameter 617
   DYNALOC installation option 617
by BMCSORT 138, 237, 251
of output data sets 87, 153
of temporary sort work data sets 138
option descriptions 153
when stacking to tape 510
dynamic allocation of output copies 410
dynamic grouping 533
dynamic sizing of output files 147

**E**

EARLYCAT option 118
EARLYRECALL option 118, 389
EATTR installation option 611
EATTR option 157, 611
EAV allocation 157, 611
elapsed time 35
electronic documentation 15
eliminating interim data sets for index keys 53
ENCIPHER option 522
encrypted copies
registration 527
Advanced Encryption Standard (AES) 521
ANSI Data Encryption Algorithm (DEA) 521
Data Encryption Standard (DES) 521
ENCIPHER option 522
key data set 522
key data set for 522, 605
KEYDSNAM installation option 522, 605
overview 521
requirements 522
Triple Data Encryption Standard (TDES) 521
ENCRIPTED option 217
ERRCONT option 602
examples
building a nonpartitioned index 415
creating a consistent database for query and review 558
creating a consistent set of copies at a point in time 582
creating an image copy without reading a space 584
creating and dynamically allocating output copies 410
drop recovery using pack dump backup 552
drop recovery when image copies are available 540
dropped table in a segmented table space 545
extracting nonpartitioned keys 415
moving a table space to another subsystem 564, 572
moving an application to another subsystem 564
OUTCOPY ONLY 582, 584
overriding installation options 429
parallel index rebuilds 441, 450
QUIESCE 558
RECOVER PLUS jobs 377
recovering a dropped table 549
recovering a dropped table space 540
recovering to a non-DB2 data set 417
recovering to a specified copy 410
recovering to a specified LRSN 413
recovering to current state 389
recovering using a change accumulation file 436
recovery plan preview 384
using LOGSCAN command 437
using SIMULATE YES option 440
using tape-stacked data sets 416
using the MAINT parameter 437
using the MAXKSORT option 441, 450
examples, simple JCL for RECOVER PLUS execution 317
EXCLUDE option
for the COPY command 187, 229
for the IMPORT command 301, 306
for the MIGRATE command 283, 287
excluding
index spaces 187, 229
RECOVERY MANAGER groups 187, 229, 283, 287, 301, 306
table spaces 187, 229, 283, 287, 301, 306
EXEC statement
in RECOVER PLUS JCL 318
utility parameters 319
execution module (AFRMAIN) 342
execution phases
descriptions 366
for table space and index recovery 373–375
for table space recovery 367, 369, 371
using Instant Snapshots 375
with BACKOUT 374
execution plan 592
EXPDT option 159
expiration date for tape copies 159
extended address volumes 69, 157, 212, 309, 611
extended format data
SHAREOPTIONS value 488
extended format sequential data set allocation 157, 611
extracted keys files work space requirements 341
extracting index keys from non-DB2 data set 249
extracting nonpartitioned keys 415

F
fallback recovery 479
FCPPRC installation option 613
features
OBID translation 37
OUTCOPY ONLY 39
running without SYSLGRNX records 39
using non-registered input copies 38
features of RECOVER PLUS 35
fixed disk storage requirements 489
FlashCopy image copies 126, 207, 217, 487, 604, 605
FLASHCOPY option 207
forward recovery without image copy 51
FROMLOGPOINT option 285, 286
description 210, 265, 269
LASTARCHQ 210, 265, 286
LASTCOPY 210, 265, 286
LASTQUIESCE 210, 265, 286
LASTSHUTDOWN 210, 265, 286
restrictions 174, 282
syntax diagram 685, 705
X’logPoint’ 210, 265, 286
FROMRBA option 286

H
hardware compression
BMCDICT table 626
hash spaces
and the BACKOUT option 483
Help
online 15
High-speed Structure Change
description 39
DSSIZE option description 217
MODEL option description 217
PIECESIZE option description 217
TRANSFORM option description 178
HISTORY installation option
BMCHIST table 628
HISTORY option
description of installation option 606
history relationship 304, 608
AUX option
description 128, 130
HISTRETEN installation option
BMCHIST table 630
HPGOBID 540, 545
HPGRBRBA 540
ICF catalog 73
IDCAMS load module 74
identifying maintenance applied 320
IMAGCOPY authority 72
image copies, invalid 479
image copy
creating 58
recovering a partitioned table space 500
SHRLEVEL CHANGE copy 479
IMPORT command
dependencies and prohibitions 300
description 90
INDEPENDENT OUTSPACE specification
syntax diagram 709
option descriptions 301
OUTCOPY specification syntax diagram 709
syntax diagram 297, 298, 709
USING specification syntax diagram 709
IMPORT command options
ANALYZE 311
CHANGE ... FROM ... TO 283, 301
CLONE 311
DEFINE 315
DSNUM 301
EXCLUDE 301, 306
INDEPENDENT OUTSPACE 308
INDEXES 301
MODEL 308
NOCOPYPEND 301
OBJECTSET 306
OUTCOPY 309
REDEFINE 313
RESET 301
REUSE 314
SYNC 301
TABLESPACE 304
UPDATE VERSIONS 314
USING specification 301, 304
IMPORT option 535
including comments 90
INCOPY option 539, 540, 555, 564, 572
description 207
object ID translation 37
restrictions 174
INCOPY specification
description 210, 217
syntax diagram 685
INCR option 207
incremental copies of nonpartitioned indexes 62
incremental image copies, cumulative 82
INDEP INTABLESPACE option 417, 555
INDEP OUTSPACE option 418, 545, 555, 558
restrictions 174
INDEPENDENT INTABLESPACE option 233, 249
INDEPENDENT OUTSPACE option
description 190, 231, 256
description for IMPORT command 308
INDEPENDENT OUTSPACE specification
description 288
syntax diagram 705, 709
INDEVT option 217
index backups
incremental copies of nonpartitioned indexes 62
index builds, multitasking 326, 333
index key sort concurrency 143, 495
index keys, extracting from non-DB2 data set 249
INDEX option 182, 185, 228, 269, 270
index rebuild 53
index sort work space requirements 338
index space status
actions for access to index space 75
initial 75
required initial status 74
unacceptable values 75
index specification
multiple indexes 185
multiple indexes for LOGSCAN 270
single index 182
single index for LOGSCAN 269
syntax diagram 685, 702
INDEX(ALL) option
restrictions 174
INDEXES NO option 283
INDEXES option 178, 265, 283, 301
INDEXES RECOVER option 283
indexes, compressed 532
INDEXLOG option
restart consideration with AUTO value 347
description for OPTIONS command option 134
description of installation option 594
INDEXSPACE option 182, 185, 269, 270
inflight resolution recovery 450
initial status restoration 78
inline copies
performance 473
restrictions 44, 212
INLINE option 207
INLOG option
description 207
restrictions 174
INLOG specification
syntax diagram 685
input copies, non-registered 38
input image copy data sets 330
INSEQNO option 217
installation options
$C30DOPT member 587
affecting restart 346
alphabetical listing 589
AMSCAT 594
AUTOSIZE 605
AUX 608
BACKOUT 609
BINDQUALIFIER 606
BMCHIST 628
CHECKINT 346, 593
CHECKPT 346, 593
CPUMIPS 613
DATACLAS 612
DATAMVR 606
DEFINE 614
DISKIORATE 613
EATTR 611
ERRCONT 602
FCPPRC 613
HISTORY 606, 628
HISTRETN 630
INDEXLOG 594
IXRECP 595
KEYDSNAM 522, 605
KSORTSHARE 607
LOCCASEL 604
LOCCPSEL 604
MAXDRIVE 599
MAXKSORT 597
MAXLOGS 599
MAXLSORT 606
OPNDB2ID 73, 593
OUTCOPY 596
PLANRECV 592
PUBLICPLAN 606
RCLTSK 594
RDB2STAT 78, 594
REMCPSN 605
RESINV 600
RICHK 615
SMCORE 599
SNAP 612
SORTDEVT 596
SORTNUM 597
STORCLAS 612
TBUFFS 592
TZRULE 602
USEHDROBIDS 610
USELOGS 614
WKUNIT 600
installation options module (AFROPTS) parameter in JCL 324
installation options module for RECOVER PLUS 587
installation options, overriding 429
installation SYSADM authority 72
Installation System 587
Instant Snapshots
  general 52, 484
  migration 486
  OBIDXLAT 486
  registration 485
  restrictions 212, 486
  SNAP phase 485
  with other BMC Software utilities 487
invalid image copies 479
INVOLUME option 217
issuing a QUIESCE 558
IXEXPAND option 532
IXRECP option 135, 429, 595

J
JCL
  checkpoint override parameter 322
  DD statements 325
  EXEC statement 318
  group attach name 320
  input image copy data sets 330
  installation options module (AFROPTS) 324
  JOB statement 318
  JOBLIB statement 325
  message level parameter (MSGLEVEL) 322
  output image copy data sets 330
  RDB2STAT override parameter 323
  REGION parameter 318
  simple example for RECOVER PLUS execution 317
  STEPLIB DD statement 325
  subsystem ID (SSID) 320
  utility identifier (UTILID) 320
job restart 83
JOB statement
  in RECOVER PLUS JCL 318
  parameters 318
  JOBLIB statement 325

jobs
  building 317
  displaying status 347
  restarting 342
  starting 341

K
key data set for encrypted copies 522, 605
contents 524
management 526
overview 522
requirements 522
key sorts, shared 146
key work data sets option restrictions 95
KEYDSNAM installation option 522, 605
keys sort work files 143, 145, 333, 597, 606
KEYSORT option 138
KSORTSHARE option
  description 146
  installation option description 607

L
L001WK DD statements 119
L001WKnn 336, 340
large object recovery 59
last copy
  migrating from 286
  recovering from 210
  scanning from 265
last shutdown
  migrating to 290
  recovering to 193
  scanning to 274
LASTARCHQ option 193, 210, 265, 274, 286, 290
LASTCOMMONQ option 193, 274, 290
LASTCOPY option 210, 265, 286, 293, 411
LASTQUIESCE option 193, 210, 265, 274, 286, 290, 413
LASTSHUTDOWN option 193, 210, 265, 274, 286, 290, 564, 572
limit keys, changed 520
limitations of RECOVER PLUS 44
LOAD LOG YES 207
LOAD LOG YES with REPLACE 38
LOB data
  BMCSYNC table considerations 635
  MEMLIMIT setting 70
LOBs
and the BACKOUT option 131, 483
exception status and LOBs 60
general recovery 59
recovery of LOG NO auxiliary spaces 59
local site copies, primary and backup 212
LOCALSITE option 201
LOCCASEL option 604
LOCCPSEL option 604
log files, reading concurrently 506
log input phase 366
log marks, migrating to 290
log marks, recovering to 150, 193
Log Master for DB2 540, 545, 549, 582
log points
migrating from 286
migrating to 290
recovering from 210
recovering to 193
scanning from 265
scanning to 274
log range table 631
log sort space requirements 340
LOGAPPLY ONLY option 200, 277, 294
logging attribute 131, 483
LOGONLY option 200, 277, 295, 552
LOGOUNnn 336, 606
LOGPOINT LOGMARK logMarkName option 150
LOGPOINT X'logpoint' option 150
logs, maximum 599
LOGSCAN command
dependencies and prohibitions 264
description 90
example 437
index space specification 269, 270
overview 47
report contents 47
syntax diagram 261
syntax diagrams 702
SYSSCAN 47
table space specification 265
LOGSCAN command options

AUX 265, 270
BACKOUT 274
CLONE 272
DSNUM 265, 269
FROMLOGPOINT 265, 269
FROMRBA 265, 269
INDEX 269, 270
INDEX (ALL) TABLESPACE 270
INDEXSPACE 269, 270
LOGAPPLY ONLY 277
LOGONLY 277
TOLOGPOINT 274
TORBA 274
LOGSORT
advantages 474
strategy 473
LOGSORT option
column option description 119, 200
syntax diagram 685
LOGSORT specification
MIGRATE command 294
syntax diagram 705
long names in SYSIN 91
LRSN, finding with DSN1LOGP utility 545
LxxxxWKnn 336, 340

M
MAINT parameter 437
MAINT restart parameter used in JCL 320
maintaining common utility tables 623
maintenance tracking 320
materialized pending changes 520
MAXDRIVE option
installation option 599
MAXDRIVES option
description 127
example 429
use with MAXLOGS 506
MAXKSORT option
description 143
example 441, 450
installation option description 597
setting 495
MAXLOGS option
command option description 127
installation option description 599
usage rules 506
use with MAXDRIVES 506
MAXLSORT option
   description 145
   installation option description 606
   setting 497
MAXPRIM option 157
MEMLIMIT parameter 83, 318
   recommended setting with LOB data 70
   recommended setting with XML data 70
   setting 70
MEMLIMIT system parameter 70
merge phase 366
MERGE phase
   parallel 145
message level parameter (MSGLEVEL), in JCL 322
MGMTCLAS option 155
MIGRATE command
   description 90
   FROMLOGPOINT syntax 705
   FROMRBA or FROMLOGPOINT specification
      syntax diagram 705
   FROMRBA syntax 705
   INDEPENDENT OUTSPACE specification
      syntax diagram 705
   LOGSORT specification syntax diagram 705
   OBJECTSET specification 705
   option descriptions 283
   syntax diagram 277, 279
   syntax diagrams 705
   TORBA or TOLOGPOINT specification syntax
      diagram 705
   USING specification syntax diagram 705
MIGRATE command options
   ANALYZE 295
   AUX 304
   CLONE 295
   DSNUM 283
   EXCLUDE 283, 287
   FROMLOGPOINT 285, 286
   FROMRBA 286
   FROMRBA or FROMLOGPOINT specification 286
   INDEPENDENT OUTSPACE 288
   INDEXES 283
   INDEXES NO 283
   INDEXES RECOVER 283
   LOGAPPLY ONLY 294
   LOGONLY 295
   LOGSORT 294
   MODEL 288
   OBJECTSET 287
   OBJECTSET specification 287
   RESET 283
   TABLESPACE 285
   TOCOPY LASTCOPY 293
   TOLOCATION 296
   TOLOGPOINT 290
   TORBA 290
   USING dataset 283
   USING specification 283
MIGRATE command
   dependencies and prohibitions 282
migrating
to a log mark 290
migrating data 558, 572
migration
   from a log point 286
   from the last archive quiesce point 286
   from the last copy 286
   from the last image copy 286
   from the last quiesce point 286
   from the last shutdown 286
   non-DB2 data set 288
   to a log point 290
   to archive quiesce points 290
   to last shutdown 290
   to quiesce points 290
migration of copies 535
MODEL option
data set symbolic variables 190, 256
for extracting indexing keys 249
for INDEPENDENT OUTSPACE specification
190, 288, 308
for rebuilding indexes 231, 233, 256
MODELDCB option 155
moving an application to another subsystem 564
MSGLEVEL option 326
multiple commands, using 92
multiple data sets
  BACKOUT 193
  DSNUM ALL 182, 269
forward recovery of index spaces 516
INDEP OUTSPACE 190, 231, 256, 288, 308
INDEXLOG AUTO 347
Instant Snapshots 485, 486
MAXKSORT 143, 597
OUTCOPY 212, 309
REDEFINE 84, 203, 241, 260, 313
using copies 500
multiple index specification 185, 270
multiple RECOVER PLUS commands
  restrictions for nonpartitioned table spaces 95
  restrictions for partitioned table spaces 95
multitasking index builds 326, 333

N
names of common utility tables, determining 625
names, long 91
naming conventions
  common repository 653
NEW restart parameter 320
NEW/RESTART restart parameter 320
NEW/RESTART(PHASE) restart parameter 320
NOCOPYPEND option 207, 242, 301
NodeID 531
NOEARLYCAT option 118
NOEARLYRECALL option 118
non-DB2 data sets
  for extracting index keys 249
  for import 308
  for migration 288
  for rebuilding indexes 231, 233, 256
  for recovery 190
non-registered copy specification, syntax diagram 685
non-registered input copies 38
nonpartitioned index recovery
restrictions 95
single and multiple jobs 469
nonpartitioned index, building 415
nonpartitioned indexes, incremental copies 62
nonpartitioned keys, extracting 415
NOSCRATCH option, for REDEFINE 203, 241, 260, 313
NOSYSLGRNG option 121, 539
NOT LOGGED attribute 131, 483
NOWORKDDN option 235, 389
NOWORKDDN strategy 477
NUMREC option 120, 239, 252

O

OBID
  translation example 545
  example use 545
  finding for dropped objects 540
  finding the value for 545
  for newly created objects 540
  representation in PGSOBD 540
  using the default for the source 564
  using the default for the target 572
OBID translation, description 37
OBIDs, use of those in header 130, 610
OBIDXLAT option 539, 540, 545, 549, 552, 555, 558, 564, 572
  description 205
  restrictions 174
  syntax diagram 685
object identifiers, obtaining 540, 545, 558, 564, 572
OBJECTSET option 182, 183, 187, 265, 269, 270, 272, 273
  restrictions 174
OBJECTSET specification
  description 187, 272, 287
  for IMPORT command 306
  syntax rules 187, 272
OBJECTSET specification syntax diagram 705
OBJECTSET specification
  IMPORT command 298, 709
  MIGRATE command 279
OBJSET_DEF table 655
OBJSET_SQL table 657
OBJSETS table 653
obtaining data definitions 540, 545, 549, 552
ON ERROR option, description 142
Online Consistent Copies
and TRANSFORM 39
and compressed indexes 532
and FlashCopy 487
and INCOPY 207
and log apply 207
and migration 555
as additional resource 42
cabinet copy support 63
DATAMVR option 64, 138
feature of Recovery Management solution 40
using 64
online Help 15
online schema evolution 518
online schema evolution support 518
operating environment 69
operating system requirements 69
OPNDB2ID installation option 73, 593
option
  EXCLUDE 187, 229, 283, 287, 301, 306
  MODEL 308
  SNAP 151
options
alphabetical listing 96, 661
ANALYZE 139, 201, 240, 253, 259, 295, 311
AUTOSIZE 147
AUX 128, 178, 185, 231, 265, 270, 304, 608
AVGRECSZ 120
BACKOUT 131, 193, 274, 609
BACKOUT AUTO 131
BUILDINDEX 256
CATLG 155
CHANGE ... FROM ... TO 283, 301
CHECKINT 122
CLONE 201, 235, 250, 258, 272, 295, 311
COMPRESSED 217
CPUMIPS 613
DATACLAS 155, 612
DATAMVR 138
DEFINE 131, 201, 238, 315, 614
DISKIORATE 613
DROPRECOVERY 178
DSNAME 153, 217
DNUM 182, 187, 229, 265, 269, 283, 301
DSSIZE 217
EARLYCAT 118
EARLYRECALL 118
EATTR 157, 611
ENCIPHER 522
ENCRYPTED 217
EXPDT 159
FCPPRC 613
FLASHCOPY 207
FROMLOGPOINT 210, 265, 269, 285, 286
FROMRBA 210, 265, 269, 286
HISTORY 130, 606
INCOPY 207, 210, 217
INCR 207
INDEPT 217
INDEPT 217
INDEX 182, 185, 228, 269, 270
INDEXES 178, 265, 283, 301
INDEXES 283
INDEXES RECOVER 283
INDEXLOG 134
INDEXSPACE 182, 185, 269, 270
INLINE 207
INLOG LOGPOINT 207
INLOG RBA 207
INSEQNO 217
ANALYZE 139
AUTOSIZE 147
AUX 128
AVGRECSZ 120
BACKOUT 131
BACKOUT AUTO 131
CHECKINT 122
DATAMVR 138
DEFINE 131
EARLYCAT 118
EARLYRECALL 118
HISTORY 130
INDEXLOG 134
IXRECP 135
KEYSORT 138
KSORTSHARE 146
LOGPOINT LOGMARK logMarkName 150
LOGPOINT X’logpoint’ 150
LOGSORT 119
MAXDRIVES 127
MAXSORT 143, 495
MAXLOGS 127
MAXSORT 145, 497
NOEARLYCAT 118
NOEARLYRECALL 118
NOSYSLGRNG 121
NUMREC 120
ON ERROR 142
OUTCOPY 127
RBALRSN_CONVERSION 245
RECOVERYPOINT 147
RESINV 125
RESOURCE SELECTION 126
RICHK 128
SIMULATE YES 141
SMCORE 125
SNAP 151
SORTDEVT 120, 139
SORTDIAG 124
SORTDYN 124
SORTNUM 120, 139
STOGROUP 123
TIMESTAMP 150
TRTCH 136
URIDDDN 149
USEACCUM 122
USEHIDROBIDS 130
USEORDER 123
WTOR 142
XBMID 136

ZIIP 137
OS requirements 69
OUTCOPY ONLY
description 39
restrictions 212
OUTCOPY option 410, 540, 552, 564, 572, 582, 584
description 127, 212
for RECOVER TABLESPACE 212
installation option description 596
OUTCOPY specification
description for IMPORT command 309
syntax diagram 709
OUTCOPY specification syntax diagram 685
OUTCOPYDDSN option
description 220
example 410
syntax diagram 298
OUTCOPYDSN option, description 220
OUTPUT command
description 87
example 410
option descriptions 151
restrictions on use 153
syntax diagrams 683
OUTPUT command options
CATLG 155
DATACLAS 155, 612
DSNAME 153
ENCIPHER 522
EXPDT 159
for disk and tape 155
for disk data sets only 157
for tape data sets only 159
MAXPRIM 157
MGMTCLAS 155
MODELCDB 155
name 153
RETPD 159
SPACE 157
STACK 159
STORCLAS 155
TRTCH 159
UNIT 155
UNITCNT 155
VOLCNT 155
VOLUMES 157
output data set
dynamic allocation 87
names in OUTPUT statement 153
output descriptor
PART option 235, 248
partitioned index recovery
  in multiple jobs 468
  restrictions 95
partitioned table space considerations 467
partitions
  added 518
  dropped 519
performance considerations 467
PGSOBD 540
phases
  descriptions 366
  of RECOVER PLUS 342
  restarts allowed 342
  SNAP 485
PIECESIZE option 217
PIT_RBA 475
PLANRECV installation option 592
point of consistency, obtaining 564, 572, 582
point-in-time recovery specification
  description 193, 274
  strategies 481
previewing recovery activity
  discussion 54
  example 384
  option description 201, 253, 259
primary copies, local and recovery site 212
processing, fallback 479
PRODREG table 658
PSID 540, 545, 558, 564, 572
PSRCP status 74
PUBLIC privilege 606
publications, related 15
PUBLICPLAN installation option 606

Q
QUIESCE 558, 564, 572, 582
quiesce points
  migrating from 286
  migrating to 290
  migration to 290
  recovering from 210
  recovering to 193
  scanning from 265
  scanning to 274
### R

**R+/CHANGE ACCUM**
- change accumulation file, using 436, 504
- utility integration 44

**RACF (IBM Resource Access Control Facility)**
- security exit 71
- RACF authority 73
- RACF ID 73
- RBA, finding with DSN1LOGP utility 545
- RBDP index 429
- RCLTSK option 594
- RDB2STAT option
  - general 78
  - installation option description 594
- RDB2STAT override parameter
  - in JCL 323
    - RDB2STAT(NO) 323
    - RDB2STAT(RO) 323
    - RDB2STAT(RW) 323
    - RDB2STAT(YES) 323
  - RDB2STAT(NO) override parameter 323
  - RDB2STAT(RO) override parameter 323
  - RDB2STAT(RW) override parameter 323
  - RDB2STAT(YES) override parameter 323
- real-time statistics (RTS), resetting for OUTCOPY 212

**rebuilding indexes**
- by partition 228, 235

**RECOVER BUILDINDEX command**
- description 89
- example 415
- option descriptions 254
- restrictions on use 93

**RECOVER BUILDINDEX command options**
- ANALYZE 259
- BUILDINDEX 256
- CLONE 258
- INDEPENDENT OUTSPACE 256
- REDEFINE 260
- REUSE 261
- SKEYDDN 259, 336
- TABLESPACE 258

**RECOVER INDEX command**
- AUX option 185
- dependencies and prohibitions 174, 175, 282
- description 88
- index space specification 182, 185
- restrictions on use 174
- syntax diagrams 162, 685, 697
- use 558, 564

**RECOVER INDEX command options**
RECOVER UNLOADKEYS command options

- ANALYZE 253
- CLONE 250
- INDEPENDENT INTABLESPACE 249
- NUMREC 252
- PART 248
- SKEYDDN 250, 335
- SORTDEVT 251
- SORTNUM 251
- TABLESPACE 248
- UNLOADKEYS 247

RECOVERDB authority 72

recovery

dropped table from a simple table space 549

dropped table in segmented table space 545

dropped table space, description 539

to a non-DB2 data set 417

to a specified copy 410

to a specified LRSN 413

to the current state 389

RECOVER UNLOADKEYS command
after limit key changes 520
rotated partitions 519
after added partitions 518
change accumulation file, using 504
dropped partitions 519
from a log point 210
from the last archive quiesce point 210
from the last copy 210
from the last image copy 210
from the last quiesce point 210
from the last copy 210
hash spaces 483
LOBs 59, 483
multiple indexes 492
multiple partitions concurrently 467
nonpartitioned indexes 95
NOT LOGGED spaces 483
output 418
partitioned indexes 95
plan 384
simulation 440
to a log mark 150, 193
to a log point 193
to archive quiesce points 193
to last shutdown 193
to quiesce points 193
using a full volume backup 81
using BUILDINDEX 53
using incremental image copies 82
using NOWORKDDN 477
using only the log 51
using SHRLEVEL CHANGE copies 479
using UNLOADKEYS using 53
using UNLOADKEYS/BUILDINDEX 478
while maintaining activity while 57
without an image copy 51
Recovery Management for DB2 33, 34, 39, 147
RECOVERY MANAGER for DB2 582
RECOVERY MANAGER groups 533
RECOVERY MANAGER utility integration 43
recovery plan, option descriptions 139, 201, 240, 253, 259
recovery simulation 55, 56, 141, 177, 227, 228, 247, 255
recovery site copies, primary and backup 212
recovery strategies
building nonpartitioned indexes on partition table space 53
compressed indexes 532
eliminating interim data set for index keys 53
fallback 479
large partitioned objects 506
large table spaces 506
LOGSORT 473
maintaining activity 57
planning recovery resources 505
previewing recovery activity 54
R+/CHANGE ACCUM, using 504
recovery with an image copy 51
recovery with an Instant Snapshot 52
resource selection 53
separate change accumulation files, using 506
separate copies, using 506
sorting the log 473
testing recovery 54
using BUILDINDEX 53
using UNLOADKEYS 53
recovery, local
BACKOUT AUTO option 51
backout recovery 49
RECOVERYDDN option
description 221
syntax diagram 685
RECOVERYDSN option, description 221
RECOVERYPOINT 60
RECOVERYPOINT option, description 147
RECOVERYSITE option 201
RECP status 429
REDEFINE option 203, 241, 260, 313
redirection of output 418
reduced elapsed time 35
REGION parameter 318
REGISTER option 212
registering index backups 61
registration in the BMXCOPY table 57
related publications 15
REMCPSSEL option 605
removing objects for restart 342
REORG LOG YES 38, 207
REPAIR VERSIONS 528
repository tables 653
requesting jobs, restarting 83
requirements
extracted keys files work space 341
fixed disk storage 337
index sort work space 338
log sort space 340
sort work space 337, 340
virtual memory 337
RESET option 283, 301
RESETRTS option 212
residing in APF library 74
RESINV option 125, 600
RESOURCE SELECTION option 126
resource selection, using 53
restart
affected by installation options 83
at a MERGE phase 342
at beginning of LOG INPUT phase 342
at the beginning of the last incomplete phase 320
considerations when creating output copies 344
during BUILD phase 342
from last incomplete phase 320, 342
from last sync point 320
from point of failure 344
installation options that affect 346
limitations 236
LOGSORT restriction 473
RECOVER PLUS job 342
removing/adding objects 342
syntax options that affect 342
table for RECOVER PLUS phases 342
transform 346
using checkpoints for restart 83
using INDEXLOG AUTO 347
using sync restart 320
utility parameters that affect 320
restart parameter
MAINT 320
NEW 320
NEW/RESTART 320
NEW/RESTART(PHASE) 320
RESTART 320
RESTART(PHASE) 320
TERM 320
restoration of initial status 78, 323
restrictions
for key work data sets 95
for multiple RECOVER commands 93, 95
for nonpartitioned table spaces 95
for partitioned table spaces 95
for REBUILD INDEX 93
for RECOVER BUILINDEX 93
for RECOVER PLUS 44
for SKEYDDN 95
for SORTDEVT 95
for SORTNUM 95
for STOGROUP-defined spaces 94
for WORKDDN 95
using LOGSORT 474
using multiple commands 93
retention period, tape copies 159
RETPD option 159
RETRY parameter of DYNALOC option 617
retrying dynamic allocation of SORTWK 617
REUSE option 204, 242, 261, 314
reusing data sets 204, 242, 261, 314
revision bars 18
RICHK option 178, 304
description of installation option 615
rotated partitions 519
RTS, resetting for OUTCOPY 212
running BMC utilities concurrently 638

S

S001WKnn 333, 336
scan
to quiesce points 274
from a log point 265
from the last archive quiesce point 265
from the last copy 265
from the last image copy 265
from the last quiesce point 265
from the last shutdown 265
to a log point 274
to archive quiesce points 274
to last shutdown 274
secondary extents, sliding-scale calculation 190, 231, 256
security
mechanisms 71
setting SMCORE installation option values 493
shared key sorts 146
SHAREOPTIONS value 488
SHRLEVEL 632
SHRLEVEL CHANGE copies used for recovery 479
SHRLEVEL of copies produced 581
shutdown
  migrating from 286
  migrating to last 290
  recovering from 210
  recovering to last 193
  scanning from 265
  scanning to last 274
SIMRBLD 227
SIMRBLD command 440
SIMRCVR 177, 228, 247, 255
SIMRCVR command 440
SIMULATE option 55
SIMULATE YES option, description 56, 141
SIMULATE YES option, example use 440
simulating recovery 440
simulation 55
simulation mode 56, 141, 177, 227, 228, 247, 255
SKEY DD statement 415
SKEYDDN option
  unloaded keys file space requirements 341
description 250, 259
  restart requirements 342
restrictions 95
sliding-scale calculation, for secondary extents 190, 231, 256
SMCORE installation option 493, 599
SMCORE option on the OPTIONS statement 125
SNAP installation option 612
SNAP option
description for OPTIONS command 151
snap phase 366
SNAP phase 485
SNAPSHOT option 207
sort concurrency 145, 497
sort message files 143, 145, 333, 336, 597, 606
sort options 124, 125
sort performance 489
sort sizes, determining 489
sort tasks
  management 367
  parallel 369, 371
sort work data set
defining 336
  options for REBUILD INDEX 237
options for RECOVER UNLOADKEYS 251
sort work space
  allocating 499
requirements 337, 340, 491
SORTDEVT option
description 120, 139, 237, 251
equency 429
general information 94
installation option description 596
restrictions 95
SORTDIAG option 124
SORTDYN option 124
sorting the log 473
SORTNUM option
description 120, 139, 237, 251
equency 429
general information 94
installation option description 597
restrictions 95
SPACE option 157, 212
specification
  EXEC statement 318
  JOB statement 318
SPUFI 540
SQL statements
deleting rows from the BMCDICT table 627
deleting rows from the BMCHIST table 631
deleting rows from the BMCSYNC table 637, 645
deleting rows from the BMCSYNC table for
  RECOVER UNLOADKEYS 637
deleting rows from the BMCXCOPY table 652
displaying BMC utilities 625
querying BMCHIST table 625
querying BMCXCOPY table 625
terminating BMC utilities 625
STACK option 159
stacked tape
  change accumulation file, using 506
  image copies, dynamic allocations 510
  mounts 330
  ordering copies 501
  strategies in using copies 501
START_RBA 475
STARTDB authority 72
starting a new RECOVER PLUS execution 320
starting a RECOVER PLUS job 341
starting a space after recovery 78
STATS
  IMPORT command 315
status
index space-unacceptable 75
displaying for jobs 347
table space-unacceptable 75
when RECOVER PLUS job is terminated 347
status, BMC utilities 625
STEPLIB DD statement 325
STOGROUP option 123
STOGROUP-defined data sets reallocating 203, 241, 313
for index spaces 73
reallocating 203, 241, 313
STOPDB authority 72
storage
fixed disk requirements 337, 489
virtual memory requirements 489
virtual, specified by REGION parameter 318
storage, virtual 70
STORCLAS installation option 612
STORCLAS option 155
strategy
LOGSORT 473
NOWORKDDN 477
UNLOADKEYS/BUILDINDEX 478
striped data
SHAREOPTIONS value 488
subsystem ID (SSID) in JCL 320
summary of product changes 18
SxxxWKnn 143, 145, 333, 336, 597, 606
symbolic variable
&PART5 510
&UNIQ or &UQ 510
symbolic variables
in copy data set names 510
with IMPORT 308
with IMPORT command 308
with MIGRATE 288
with MIGRATE command 288
with OUTPUT command 153, 155
with RECOVER INDEX 190
with RECOVER INDEX command 217
with RECOVER INDEXSPACE 190
with RECOVER INDEXSPACE command 217
with RECOVER TABLESPACE 190
with RECOVER TABLESPACE command 217
SYNC option 301
syntax diagram
IMPORT command 709
syntax diagrams
FROMLOGPOINT specification 685, 705
FROMMRB specification 685, 705
IMPORT command 297, 298
INCOPY specification 685
INDEPENDENT OUTSPACE specification 685, 705, 709
index specification 162, 685, 697, 702
LOGSCAN command 261, 702
LOGSORT specification 679, 685, 705
MIGRATE command 277, 279, 705
non-registered copy or INLOG 685
OBIDXLAT specification 685
OBJECTSET specification 279, 298, 705, 709
OPTIONS command 679
OUTCOPY specification 685, 709
OUTCOPYDDN specification 298, 685
OUTPUT command 683
point-in-time recovery specification 685
REBUILD INDEX command 697
RECOVER INDEX command 685
RECOVER INDEXSPACE command 685
RECOVER TABLESPACE command 685
RECOVER UNLOADKEYS command 700
RECOVERYDDN specification 685
SORT specification 679
table space specification 685, 702
TOCOPY specification 685
TOLOGPOINT specification 685, 705
TORBA option 685
TORBA or TOLOGPOINT specification 705
USING specification 705, 709
SYSADM authority 72
SYSCOPY table 549
SYSCTRL authority 72
SYSERR 326
SYSIN data set 326
SYSLGRNX 39
SYSLGRNX records, running without 39
SYSLGRNX table 545, 549
SYSOUnnn 143, 145, 326, 333, 597
SYSOUnnn data set 326
SYSPICK data set 326, 384
SYSPRINT data set 326
SYSLGRNX data set 47, 326
system (RACF) authority 73
&UNIQ or &UQ variable 510
system clock variable 510
system DBADM authority 72
system-based versioning 128, 304, 608
system-maintained temporal tables 128, 304, 608
SYSUDUMP 326
SYSUT1 DD statement
  and key work data sets 236
  eliminating interim data sets 53
  with key work data sets 333

T

table names, determining 625
table space recovery in multiple jobs 468
table space specification syntax diagram 685, 702
table space status
  actions for access to table space 75
  initial 75
  overview 74
  unacceptable values 75
  when terminating a RECOVER PLUS job 347
table space, partitioned 467
tables
  BMCSYNC 320
  BMCUTIL 320
tables, BMC
  backing up 623
  BMCHIST 628
  BMCLGRNX 631
  BMCSYNC 632
  BMCTRANS 640
  BMCUTIL 642
  BMCXCOPY 645
  considerations 623
  determining names 625
  querying 625
  warnings 623
tables, BMC Common DB2 repository
  GROUPAUTH 658
  GRPOPTS 658
  OBJSET_DEF 655
  OBJSET_SQL 657
  OBJSETS 653
  PRODREG 658
tables, BMCDICT 626
TABLESPACE option 178, 231, 248, 258, 265, 285, 304
TAG option 212
tape copies
  expiration date 159
  retention period 159
Tape Mount Management 159
tape mounts, stacked 330
tape pick list 54
tape stacking example 416
TBUFFS option 592
TDES (Triple Data Encryption Standard) 521
temporal tables 128, 130, 304, 608
TERM restart parameter 320
terminating BMC utilities 625
termination
  RECOVER PLUS job 347
termination, RECOVER PLUS job 320
testing recovery strategy 39, 55
TIMESTAMP option 150, 217
timestamp recovery 147, 450
TMM (Tape Mount Management) 159
TOCOPY LASTCOPY option 293
TOCOPY option
  description 197
  example 410
  syntax diagram 685
TOCOPY specification
  description 197
  LASTCOPY option 293
  restrictions 174, 282
TOLOCATION option 296
TOLOGPOINT LOGMARK logMarkName 193, 290
TOLOGPOINT option 413, 540, 545, 558
  BACKOUT 193, 274
  description 193, 274
  LASTARCHQ 193, 274, 290
  LASTCOMMONQ 193, 274, 290
  LASTQUIESCE 193, 274, 290
  LASTSHUTDOWN 193, 274, 290
  X'logPoint' 193, 290
  X\xd5 logPoint\xd5 274
TOLOGPOINT specification
  syntax diagram 705
TORBA LOGMARK logMarkName 193, 290
TORBA option 290, 413, 540, 545, 558
  BACKOUT 193, 274
  description 193, 274
  LASTARCHQ 193, 274, 290
  LASTCOMMONQ 193, 274, 290
  LASTQUIESCE 193, 274, 290
  LASTSHUTDOWN 193, 274, 290
  X'logPoint' 193, 290
  X\xd5 logPoint\xd5 274
TORBA or TOLOGPOINT specification
  for the MIGRATE command 290
TORBA specification
  syntax diagram 705
TORBA/TOLOGPOINT specification
syntax diagram 685
tracking maintenance applied 320
TRANSFORM option 39, 178
transform, restarting 346
Triple Data Encryption Standard (TDES) 521
TRTCH option 136, 159
TZRULE option 602

U
UNIT option 155
UNITCNT option 155
unload phase 366
unloading indexes by partition 248
UNLOADKEYS entries, cleaning up 637
UNLOADKEYS option
and indexes defined using an expression 247
description 247
UNLOADKEYS/BUILDINDEX strategy 478
UPDATE VERSIONS option 204, 314
URIDDN option 149
USEACCUM option 122, 436
USEHDROBIDS installation option 610
USEHDROBIDS option 130
USELOGS installation option 614
USEORDER option 123
USING specification description 283, 301, 304
USING specification syntax diagram 705, 709
utility identifier (UTILID) in JCL 320
utility integration
COPY PLUS 43
DB2 resources 42
overview 42
R+/CHANGE ACCUM 44
RECOVERY MANAGER 43
utility parameter
checkpoint override parameter 322
group attach name 320
installation options module (AFROPTS) 324
message level 322
on the EXEC statement 319
RDB2STAT 78
RDB2STAT override parameter 323
restart parameter 320
subsystem ID (SSID) 320
utility identifier (UTILID) 320

V
VCAT-defined data sets
option ignored (REDEFINE) 241, 260
reallocating 203, 313
version information 528
versioning relationship 128, 304, 608
versioning, steps to handle 528
virtual memory requirements 337, 489
virtual storage 70
virtual storage used by RECOVER PLUS 318
VOL=SER parameter 344
VOLCNT option 155
VOLUMES option 157
VSAM data sets
reallocating 203, 313
and REDEFINE NO 241

W
WKUNIT installation option 600
work data set ddname 235
work data sets, defining for sorts 336
work file 415
work space requirements 340
WORKDDN option
description 235
restrictions 95
working with ASCII data 488
WTOR installation option 602
WTOR option on the OPTIONS command 142

X
XBM subsystem ID 136, 600
XBMID option 136, 600
XML data
BMCSYNC table considerations 635
MEMLIMIT setting 70
XML objects
DocID 531
NodeID 531
recovering 531
Z

ZIIP enablement 601
ZIIP installation option 601
ZIIP option

and the OPTIONS statement 137
and the XBMID option 137
software requirements 71