LOADPLUS for DB2
Reference Manual

Supporting

Version 11.2 of LOADPLUS for DB2
Version 11.2 of BMC High Speed Utilities for DB2
Version 11.2 of BMC Object Administration for DB2
Version 11.2 of BMC Database Administration 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. From this website, you can obtain information about the company, its products, corporate offices, special events, and career opportunities.

United States and Canada

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

Outside United States and Canada

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


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.

BatchPipes, DB2, DB2 Universal Database, DRDA, FlashCopy, IBM, IBM z, IMS, Language Environment, MVS, RACF, RETAIN, System z10, z/Architecture, z/OS, and z10 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 UNPUBLISHED—RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES.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. 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.
## LOADPLUS about this book
- Related publications: 13
- Conventions: 14
- Syntax diagrams: 14
- LOADPLUS summary of changes: 16

## Chapter 1 Introduction to LOADPLUS
- Overview of LOADPLUS: 25
- BMC Software solution integration: 26
- Tasks that LOADPLUS performs: 28
- Differences between LOADPLUS and the IBM DB2 LOAD utility: 33
- How LOADPLUS works: 38
  - LOADPLUS processing phases: 38
  - LOADPLUS data sets: 45
- Associated products and common components that LOADPLUS uses: 47

## Chapter 2 Operational considerations in LOADPLUS
- System setup for LOADPLUS: 49
  - DB2 support in LOADPLUS: 49
  - LOADPLUS system and software requirements: 49
  - Required authorizations for LOADPLUS: 50
  - Overriding the MEMLIMIT system parameter: 55
  - Customizing products that prevent x37 abends in LOADPLUS: 55
- Number of DB2 threads that LOADPLUS uses: 56
- Load jobs that invoke DSNUTILB: 57
  - LOADPLUS features that require DSNUTILB: 57
  - Enabling LOADPLUS to invoke DSNUTILB: 58
  - General restrictions for a DSNUTILB load: 59
  - Input file for a DSNUTILB load: 59
  - Data set allocation for a DSNUTILB load: 60
  - INTO TABLE statement for a DSNUTILB load: 61
  - Considerations for other LOADPLUS options when running a DSNUTILB load: 62
- LOADPLUS serialization and concurrency: 66
  - Executing BMC utilities concurrently: 66
  - Object status when running LOADPLUS: 68
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>How LOADPLUS handles invalid value errors when generating identity</td>
<td>113</td>
</tr>
<tr>
<td>column values</td>
<td></td>
</tr>
<tr>
<td>Additional identity column considerations</td>
<td>113</td>
</tr>
<tr>
<td>XML data</td>
<td>114</td>
</tr>
<tr>
<td>Loading XML data natively</td>
<td>114</td>
</tr>
<tr>
<td>Considerations for loading XML data natively</td>
<td>115</td>
</tr>
<tr>
<td>Loading XML data by invoking DSNUTILB</td>
<td>118</td>
</tr>
<tr>
<td>LOB data</td>
<td>119</td>
</tr>
<tr>
<td>Loading LOB data natively</td>
<td>119</td>
</tr>
<tr>
<td>Considerations for loading LOB data natively</td>
<td>120</td>
</tr>
<tr>
<td>Loading LOB data by invoking DSNUTILB</td>
<td>122</td>
</tr>
<tr>
<td>Unicode data</td>
<td>123</td>
</tr>
<tr>
<td>Loading Unicode data</td>
<td>123</td>
</tr>
<tr>
<td>Considerations when loading Unicode data</td>
<td>124</td>
</tr>
<tr>
<td>CSV data</td>
<td>126</td>
</tr>
<tr>
<td>Loading CSV data</td>
<td>126</td>
</tr>
<tr>
<td>CSV data format</td>
<td>127</td>
</tr>
<tr>
<td>Installation option considerations when loading CSV data</td>
<td>128</td>
</tr>
<tr>
<td>FORMAT CSV restrictions</td>
<td>128</td>
</tr>
<tr>
<td>CSV field specifications</td>
<td>128</td>
</tr>
<tr>
<td>Data and control cards from UNLOAD PLUS</td>
<td>132</td>
</tr>
<tr>
<td>Using data from UNLOAD PLUS in internal BMC format</td>
<td>132</td>
</tr>
<tr>
<td>Using control cards from UNLOAD PLUS after migrating data in a</td>
<td>138</td>
</tr>
<tr>
<td>standard DB2 format</td>
<td></td>
</tr>
<tr>
<td>Considerations for BatchPipes input</td>
<td>139</td>
</tr>
<tr>
<td>Considerations for table space compression</td>
<td>140</td>
</tr>
<tr>
<td>Considerations when using table space compression</td>
<td>140</td>
</tr>
<tr>
<td>Considerations when building a dictionary</td>
<td>141</td>
</tr>
<tr>
<td>Considerations when keeping a dictionary</td>
<td>142</td>
</tr>
<tr>
<td>Indexes on expression</td>
<td>142</td>
</tr>
<tr>
<td>Considerations for reordered row format</td>
<td>143</td>
</tr>
<tr>
<td>Image copies and recoverability of the loaded table space</td>
<td>145</td>
</tr>
<tr>
<td>Referential and check constraints</td>
<td>146</td>
</tr>
<tr>
<td>Referential constraints and informational referential constraints</td>
<td>146</td>
</tr>
<tr>
<td>Considerations for check constraints</td>
<td>148</td>
</tr>
<tr>
<td>Considerations for CHKP status</td>
<td>150</td>
</tr>
<tr>
<td>DB2 user exits with LOADPLUS</td>
<td>152</td>
</tr>
<tr>
<td>DB2 real-time statistics in LOADPLUS</td>
<td>152</td>
</tr>
<tr>
<td>Return code hierarchy in LOADPLUS</td>
<td>158</td>
</tr>
<tr>
<td>Image copies after recovery in LOADPLUS</td>
<td>160</td>
</tr>
<tr>
<td>XBM and SUF considerations in LOADPLUS</td>
<td>160</td>
</tr>
</tbody>
</table>
Considerations for using XBM or SUF with LOAD RESUME YES
SHRLEVEL REFERENCE ................................................................. 161
Considerations for using XBM or SUF to enforce referential constraints 161
Using XBM or SUF to enable zIIP processing .................................. 161
DB2 features that LOADPLUS does not support ............................... 162

Chapter 3 Syntax of the LOAD command 163
Command syntax rules for LOADPLUS ............................................. 163
Specifying object names in your LOADPLUS syntax ....................... 164
Alphabetical listing of LOADPLUS command options ...................... 164
LOADPLUS command syntax diagram .......................................... 168
LOADPLUS command options ....................................................... 182
  Basic LOADPLUS processing options ......................................... 183
  Statistics options ..................................................................... 288
  Copy options .......................................................................... 294
  INTO TABLE options .............................................................. 304
  Data type keywords ............................................................... 337
  Supported external DATE TIME and TIMESTAMP formats ............ 356
  Supported data type conversions in LOADPLUS ....................... 358
  Additional input field length information .................................... 375
  Sample data conversions ......................................................... 376
  Additional numeric fields considerations ..................................... 379
  Additional date time and timestamp fields considerations ............ 382
  Additional VARCHAR VARGRAPHIC VARBINARY ROWID and XML
  fields considerations .............................................................. 383
  Data translation ..................................................................... 383
  Dynamic allocation options ...................................................... 393

Chapter 4 Building and executing LOADPLUS jobs 425
Building the LOADPLUS job .......................................................... 425
  JOB statement ....................................................................... 425
  LOADPLUS EXEC statement .................................................... 426
  REGION parameter .................................................................. 426
  Utility parameters on the LOADPLUS EXEC statement .............. 427
  LOADPLUS STEPLIB DD statement .......................................... 431
  LOADPLUS DD statements ....................................................... 432
  ANALYZE option for estimating data set allocation in LOADPLUS 465
Running LOADPLUS jobs ............................................................. 466
  Invoking LOADPLUS ............................................................. 467
  Restarting LOADPLUS ........................................................... 467
  Terminating or canceling a LOADPLUS job ............................... 477
Recovering the DB2 object after terminating or canceling a LOADPLUS job .........................................................................................................................478

Chapter 5         Examples of LOADPLUS jobs 483
Overview of LOADPLUS examples .................................................................483
Example 1 LOAD REPLACE of a segmented table space ..................................487
Example 2 LOAD REPLACE of a partitioned table space ...............................499
Example 3 - Single-phase LOAD RESUME YES of a partitioned table space ..507
Example 4 LOAD REPLACE of three tables ..................................................514
Example 5 - LOAD REPLACE with data conversion .................................520
Example 6 LOAD REPLACE with multiple INTO statements ....................527
Example 7 LOAD RESUME NO using DISCARDS options and user-defined return codes ...................................................................................................................533
Example 8 LOAD REPLACE with ANALYZE PAUSE ..................................538
Example 9 LOAD RESUME YES replacing individual partitions and updating indexes ........................................................................................................551
Example 10 LOAD RESUME NO of table with check constraints ..............557
Example 11 LOAD REPLACE with dynamic allocation ..............................562
Example 12 LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY of a partitioned table space .................................................................569
Example 13 LOAD REPLACE from CSV input ...........................................583
Example 14 LOAD REPLACE from UNLOAD PLUS data in internal format ...588
Example 15 LOAD REPLACE ignoring indexes ..........................................596
Example 16 LOAD RESUME YES replacing individual tables ..................602
Example 17 Loading into a Unicode table space from EBCDIC input ..........607
Example 18 LOAD REPLACE enforcing referential constraints ..................612
Example 19 LOAD REPLACE SHRLEVEL REFERENCE using FASTSWITCH .619
Example 20 LOAD RESUME YES copying multiple partitions to a single copy data set ........................................................................................................626
Example 21 Loading LOB and XML data ....................................................632
Example 22 Running a DSNUTILB load ......................................................640

Chapter 6         Tuning LOADPLUS jobs 645
Tuning for performance in LOADPLUS .......................................................645
Setting installation options for optimal performance in LOADPLUS .........645
Specifying command options for optimal performance in LOADPLUS ...648
Additional performance information for LOADPLUS installation and command options .................................................................651
Enabling multitasking for performance in LOADPLUS ............................670
Additional performance tuning recommendations for LOADPLUS ..........672
Performance tuning for specific scenarios in LOADPLUS ..........................674
Interpreting performance-related messages from LOADPLUS .......... 679
Tuning to improve memory use in LOADPLUS .................................................. 686
   Memory requirements of the ORDER options ........................................... 687
   Tuning memory use in sort processing ....................................................... 687
   Using multitasking to improve memory use .............................................. 687
   Tuning memory use in copy processing .................................................... 687
   Tuning for availability .................................................................................. 688
      Specifying SHRLEVEL for availability .................................................. 688
      Granting data set authorization for availability ...................................... 688
      Availability when adding a relatively small amount of data ............... 688
LOADPLUS processing phases ............................................................... 689
   LOADPLUS architecture ......................................................................... 689
   ANALYZE phase in LOADPLUS ............................................................... 690
   PRELOAD phase ...................................................................................... 691
   LOAD phase ............................................................................................. 693
   COMBINED phase for a single-phase load .............................................. 695
   COMBINED phase for SQLAPPLY ........................................................... 698

Appendix A  LOADPLUS installation options 703
Overview of LOADPLUS installation options ........................................... 703
Basic LOADPLUS installation options ...................................................... 704
   Descriptions of basic LOADPLUS installation options ....................... 712
Dynamic allocation installation options for LOADPLUS ...................... 760
   Descriptions of dynamic allocation installation options for LOADPLUS . 762
DYNALOC installation option .................................................................... 783

Appendix B  Common utility tables 787
Overview of common utility tables .......................................................... 787
   Warnings and considerations for common utility tables ....................... 789
   Managing common utility tables ............................................................ 791
BMCDICT table ......................................................................................... 792
   BMCDICT table considerations ............................................................. 793
   Maintaining the BMCDICT table .......................................................... 793
BMCHIST table .......................................................................................... 794
   BMCHIST table considerations for COPY PLUS .................................. 796
   BMCHIST table considerations for RECOVER PLUS ......................... 796
   Maintaining the BMCHIST table .......................................................... 797
BMCLGRNX table ...................................................................................... 797
BMCSYNC table ........................................................................................ 798
   BMCSYNC table considerations .......................................................... 801
   Maintaining the BMCSYNC table ........................................................ 803
Cleaning up RECOVER UNLOADKEYS entries ............................................. 803
Shared access levels of BMC utilities .......................................................... 804
BMCTRANS table ...................................................................................... 806
BMCUTIL table ......................................................................................... 808
Maintaining the BMCUTIL table ................................................................. 811
BMCXCOPY table ...................................................................................... 811
Maintaining the BMCXCOPY table .............................................................. 818

Appendix C RULES installation option 819
Overview of the RULES installation option .................................................. 819
Comparison operators allowed .................................................................. 819
WHEN, NULLIF, and DEFAULTIF constants handling ................................. 820
WHEN, NULLIF, and DEFAULTIF processing order ...................................... 821
Error conditions with RULES=STANDARD ................................................ 822
  Invalid operator .................................................................................... 822
  Invalid constant in predicate ................................................................. 822
Error conditions with RULES=BMC ......................................................... 823
  Invalid constant in WHEN statement .................................................... 823
  Invalid constant in NULLIF statement .................................................. 823
  Preventing these errors ........................................................................ 824

Appendix D LOADPLUS user exits 825
Overview of LOADPLUS user exits ............................................................. 825
Requirements and restrictions for LOADPLUS user exits ......................... 825
Using sample LOADPLUS user exits ......................................................... 826
Return codes for LOADPLUS user exits .................................................... 827
Rules for the user-defined variables created in LOADPLUS user exits ........ 827
LOADPLUS Assembler user exit ................................................................. 828
  Exit parameter block DSECT ................................................................. 829
  Variable mapping block DSECT ............................................................ 831
  Sample Assembler user exit ................................................................. 832
LOADPLUS COBOL II and LE COBOL user exit ........................................ 839
  COBOL II and LE COBOL exit parameter record ................................... 839
  COBOL II and LE COBOL variable mapping record ............................... 841
  Sample COBOL II and LE COBOL user exit ......................................... 842
LOADPLUS C user exit ............................................................................ 847
  C exit parameter structure ................................................................... 847
  C exit variable mapping structure ....................................................... 848
  Sample C user exit .............................................................................. 849
LOADPLUS LE C user exit ....................................................................... 854
  LE C exit parameter structure ............................................................. 854

Contents 11
LOADPLUS about this book

This book contains detailed information about the LOADPLUS for DB2 product and is intended for IBM DB2 system administrators, DB2 database administrators, and DB2 application programmers.

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

- IBM DB2 Universal Database for z/OS (DB2) DBMS
- IBM 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.

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, database/Name).

**LOADPLUS summary of changes**

This topic summarizes product changes and enhancements by version number and release date.

**Version 11.2.00, May 2015**

This version fixes known problems in the product. In addition, this version includes the following enhancements and changes:

- LOADPLUS now loads inline LOB data natively. (Earlier versions invoked DSNUTILB.) For more information, see “LOB data” on page 119.

  **Note**
  
  PTFs BPU6680 and BPU6692 provided this functionality for version 11.1.00. For more information, see the technical bulletin dated October 31, 2014.

- You can now load LOB and XML data, including inline LOB data, to a standard VBS data set. To enable this feature, you specify the FORMAT SPANNED YES option. For more information, see “SPANNED” on page 201.

  **Note**
  
  PTFs BPU6680 and BPU6692 provided this functionality for version 11.1.00. For more information, see the technical bulletin dated October 31, 2014.

- LOADPLUS no longer requires or supports the individual installation options for each feature that requires invoking DSNUTILB. However, you must still specify DSNUTILB=YES when the feature requires it.

  This enhancement makes the following installation options obsolete:
  
  — FILEVAR
  — HASHAX
  — IXONEX
  — IXRANDBATCH
  — IXRANDBATCH Path
If your installation options module contains any of these options, the module compiles successfully but completes with return code 4 and a message indicating that the option is obsolete.

■ LOADPLUS now supports indexes in which updated entries remain in their original position on the page.

■ LOADPLUS now supplies a default value of (1950,2049) for the CENTURY installation option. (Earlier versions were not shipped with a value for this option, requiring that you supply it during installation.)

If you currently use a default value other than (1950,2049), either change the value in your installation options module after installation, or use the Installation System to migrate your current option values.

■ Starting with this release, LOADPLUS does not support DB2 Version 9. Earlier releases will continue to support Version 9.

In addition, LOADPLUS supports DB2 Version 10 only in new-function mode.

■ This version of LOADPLUS has the following changes to minimum requirements. For full requirement information, see “System setup for LOADPLUS” on page 49.

— DB2 Solution Common Code (SCC) version 11.1.00 with PTF BPJ0944
— DB2 Utilities Common Code (D2U) version 11.2.00
— BMC Common Statistics component version 11.2.00

**Version 11.1.00, January 2014**

This version fixes known problems in the product. In addition, this version includes the following enhancements and changes:

**DB2 Version 11 support - native support**

LOADPLUS supports the following features of DB2 Version 11 natively:

■ Catalog and directory changes and restructuring
Extended relative byte addresses (RBAs) and log record sequence numbers (LRSNs)
LOADPLUS supports the extended values but does not convert values that are in basic format to extended format.

Persistent read only (PRO) status
LOADPLUS terminates when any object that is participating in the load is in PRO status.

DB2 Version 11 support - invoking DSNUTILB
LOADPLUS supports the following features by invoking DSNUTILB. For more information about requirements and restrictions when using features that invoke DSNUTILB, see “Load jobs that invoke DSNUTILB” on page 57.

Indexes that are defined with EXCLUDE NULL KEYS (supported by a new installation option, UTILB_NULLIX)

Columns that contain a CCSID specification (supported by a new installation option, UTILB_COLCCSID)

Additional enhancements and changes

You no longer need to specify INLOB=YES to enable LOADPLUS to invoke DSNUTILB to load inline LOB data. This change renders the INLOB installation option obsolete. If your installation options module contains the INLOB option, the module compiles successfully but completes with return code 4 and a message indicating that the option is obsolete.

LOADPLUS no longer requires that you have a unique index defined on the row ID column of the base table when loading LOB data.

LOADPLUS no longer requires that you specify an EXTENDED BUFFER MANAGER (XBM) subsystem for load jobs that use snapshot technology. When the following conditions exist, LOADPLUS searches for an XBM subsystem that is at the correct maintenance and enablement level:

— You are performing a function that requires snapshot technology.
— A value is not in effect for the XBMID option.

In most cases, LOADPLUS natively supports loading temporal tables when you specify INDEX UPDATE. However, LOADPLUS invokes DSNUTILB when all of the following conditions exist:

— You specify INDEX UPDATE.
— A participating index is defined with BUSINESS_TIME WITHOUT OVERLAPS.

— SHRLEVEL CHANGE SQLAPPLY is not in effect.

- This version of LOADPLUS enhanced the locking technique for sequential file access to reduce the possibility of user 3900 abends.

- This version of LOADPLUS has the following changes to minimum requirements.

  — IBM System z10 processor
  
  — DB2 Solution Common Code (SCC) version 11.1.00 with PTF BPJ0689
  
  — DB2 Utilities Common Code (D2U) version 11.1.00
  
  — XBM or SUF version 6.1.00

- The LOADPLUS documentation has been updated to correct the usage of XBM or its SNAPSHOT UPGRADE FEATURE (SUF) for the LOAD RESUME YES SHRLEVEL REFERENCE feature. This feature uses the Instant Snapshot technology of XBM or SUF, not the hardware snapshot capability of XBM or SUF. The type of snapshot that LOADPLUS uses affects how you set up your XBM options to take advantage of this LOADPLUS feature.

**Version 10.2.00, June 2013**

This version fixes known problems in the product. In addition, this version includes the following enhancements and changes:

Native support for objects for which LOADPLUS invoked DSNUTILB in earlier releases

**Note**

LOADPLUS ignores a DSNUTILB YES specification for natively supported features (unless some other feature requires it). BMC recommends that you review existing jobs to ensure that you obtain the expected results. For example, LOADPLUS ignores certain keywords when invoking IBM DSNUTILB, but will use those keywords now that DSNUTILB is not being invoked.

The following enhancements provide native support for which LOADPLUS invoked DSNUTILB in earlier releases:

- In most cases, LOADPLUS natively loads temporal tables. (LOADPLUS continues to invoke DSNUTILB when you specify INDEX UPDATE while loading temporal tables.)

  This support changes the following options:
— The PERIODOVERRIDE option is now valid for both native LOADPLUS jobs and DSNUTILB load jobs. This option tells LOADPLUS to load column values from an input file into a system period start or end column that is defined as GENERATED ALWAYS.

— The TRANSIDOVERRIDE option is now valid for both native LOADPLUS jobs and DSNUTILB load jobs. This option tells LOADPLUS to load column values from an input file into a transaction start column that is defined as GENERATED ALWAYS.

■ LOADPLUS natively supports XML binary data. This support includes the ability to natively load XML data from a BLOB referenced file (by using the BLOBF keyword on the CHAR or VARCHAR data type).

The AVGSIZE keyword on the CHAR or VARCHAR data type and the XMLAVGSIZE global option now apply when you are loading XML data from a BLOB referenced file.

■ LOADPLUS natively supports user-defined XML indexes for which the indexed values are stored as SQL DATE or TIMESTAMP values.

■ LOADPLUS natively loads clone tables and base tables that participate (or have participated) in a clone relationship. Base table spaces are now supported regardless of instance number.

As part of this support, LOADPLUS has added the EXCHANGE syntax option to enable you to exchange the base and clone objects after the load process completes.

This change makes the CLONE installation option obsolete. If your installation options module contains the CLONE option, the module compiles successfully but completes with return code 4 and a message indicating that the option is obsolete.

■ LOADPLUS natively supports timestamp columns that are defined with a precision (number of microseconds) other than 6.

This change makes the TSPREC installation option obsolete. If your installation options module contains the TSPREC option, the module compiles successfully but completes with return code 4 and a message indicating that the option is obsolete.

■ LOADPLUS natively supports TIMESTAMP WITH TIME ZONE data. This enhancement includes support for both of the following capabilities:

— Loading TIMESTAMP WITH TIME ZONE input data

— Loading columns that are defined as TIMESTAMP WITH TIME ZONE

As part of this enhancement, the definition for the IMPLICIT_TZ option has changed. LOADPLUS now uses IMPLICIT_TZ natively. Use IMPLICIT_TZ to
specify the implicit time zone to use when loading a TIMESTAMP WITH TIME ZONE column with timestamp values that do not contain a time zone.

This change makes the TSTZ installation option obsolete. If your installation options module contains the TSTZ option, the module compiles successfully but completes with return code 4 and a message indicating that the option is obsolete.

■ LOADPLUS natively supports indexes that contain a key that is derived from certain expressions.

LOADPLUS currently supports expressions that use the following functions:

— UPPER
— LOWER
— SUBSTR
— DATE
— DAY
— MONTH
— YEAR

Note

PTFs BPU3684, BPJ0470, and BPJ0507 provided this functionality for version 10.1.00. For more information, see the technical bulletin dated April 6, 2012.

■ LOADPLUS natively supports indexes that contain non-key columns.

This change makes the IXINCLCOL installation option obsolete. If your installation options module contains the IXINCLCOL option, the module compiles successfully but completes with return code 4 and a message indicating that the option is obsolete.

■ LOADPLUS natively loads universal table spaces that are defined as MEMBER CLUSTER.

This change makes the UTSMEM installation option obsolete. If your installation options module contains the UTSMEM option, the module compiles successfully but completes with return code 4 and a message indicating that the option is obsolete.

**Additional enhancements and changes**

■ You can now tell LOADPLUS to cancel threads in DB2 that might prevent a successful drain. A new command option, FORCE, enables this functionality;
corresponding installation options FORCE, FORCE_AT, and FORCE_RPT provide default values for this option.

This option requires additional DB2 authorization.

You can also specify a new DD statement, BMCFORCE DD, to contain the thread cancelation report.

- **LOADPLUS** now supports LOAD RESUME YES SHRLEVEL REFERENCE. This option uses the Instant Snapshot capability of the BMC EXTENDED BUFFER MANAGER (XBM) product or its SNAPSHOT UPGRADE FEATURE (SUF) technology.

  Also, the SHRLEVEL installation option has changed. You can now specify a value of REFERENCE for either parameter of this option.

- **LOADPLUS** now supports LOAD RESUME YES SHRLEVEL CHANGE (without the SQLAPPLY keyword) for PART \( n \) REPLACE.

  **Note**

  When you specify LOAD RESUME YES SHRLEVEL CHANGE without PART \( n \) REPLACE, LOADPLUS runs an SQLAPPLY load as it did in earlier versions.

  Also, the SHRLEVEL installation option has changed. You can now specify a value of CHANGE for either parameter of this option.

- **LOADPLUS** has enhanced statistics collection:

  - For UPDATEDB2STATS YES, BMCSTATS YES, and BMCSTATS REPORT, LOADPLUS now collects statistics for the following objects:

    - LOB table spaces

    - Partition-by-growth partitions that were added during the load job

  - For UPDATEDB2STATS YES, LOADPLUS now updates additional DB2 catalog statistics by passing the following options to the Common Statistics component:

    - UPDATEDB2 ALL (instead of UPDATEDB2 ACCESSPATH)

    - HISTORY ALL (instead of HISTORY ACCESSPATH)

    - KEYCARD Y (instead of KEYCARD N)

  For information about how these options affect which statistics are updated in the DB2 catalog, see the DASD MANAGER PLUS documentation.

  **Note**

  You cannot change these statistics options.
You can now send LOADPLUS output messages to a second output data set by specifying a SYSPRIN2 DD statement in your JCL. In a worklist environment, you can use SYSPRIN2 to view LOADPLUS output in real time.

**Note**
PTF BPU3886 provided this functionality for version 10.1.00.

LOADPLUS fully supports compressed indexes that have been versioned.

**Note**
PTFs BPU4280 and BPJ0506 provided this functionality for version 10.1.00. For more information, see the technical bulletin dated April 6, 2012.

You can now specify the type of data set that you want LOADPLUS to create during dynamic allocation.

When you specify the &JDATE variable with the DSNPAT option and you are running a DSNUTILB job, LOADPLUS now translates this variable to the IBM &JDATE(3,5) variable.

This release removes the ACFORTSS option. Now, if the following conditions exist, you must ensure that the DB2 external security exit (DSNX@XAC) from Computer Technologies is implemented:

— You use the Computer Technologies CA-ACF2 or CA-Top Secret security product for DB2.

— You previously used ACFORTSS=Y.

This version of LOADPLUS has the following changes to minimum requirements.

— BMCSORT version 2.4.01
— DB2 Solution Common Code (SCC) version 11.1.00
— DB2 Utilities Common Code (D2U) version 10.2.00
— High-speed Apply Engine version 11.1.00 (for SQLAPPLY load)
— BMC Common Statistics component version 11.1.00 (to update DB2 catalog statistics or the DASD MANAGER PLUS database statistics)

Starting with this release, LOADPLUS does not support DB2 Version 8. Earlier releases will continue to support Version 8. Future releases of LOADPLUS will not support the following modes:

— DB2 Version 9 CM
This release includes the following documentation changes:

— 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 located in the following books:

— *Installation System User Guide*

— *BMC Products and Solutions for DB2 Configuration Guide*
Introduction to LOADPLUS

This chapter describes the features and advantages of LOADPLUS and provides an architectural overview of the product.

Overview of LOADPLUS

The BMC Software utility products for DB2 address the needs of database administrators (DBAs) and system administrators by providing high-performance database administration and utility products. The LOADPLUS for DB2 product is a BMC Software product for loading DB2 data.

DB2 is a powerful relational database management system whose wide success in the DBMS community rests on its ability to facilitate fast application development and provide easy access to data. As users depend more and more on the DB2 DBMS for critical business applications, the need for continuous operations becomes crucial.

The need to load data

In a DB2 environment, there often is a need to load large amounts of data into DB2 tables. Data is loaded to initially populate a table, periodically replace the data in a table, add data to a table, or reload data into a table.

This data can originate from sources such as:

- Other DB2 tables (possibly from a different DB2 system)
- Other DBMSs, including IBM IMS or distributed-systems DBMSs
- Applications that generate sequential files

In many cases, users load millions of rows. Because data in the table space is often unavailable to applications during the load process, the load process must be completed as quickly as possible.
**The LOADPLUS solution**

BMC developed a product line to respond to the needs of DBAs, system administrators, and other DB2 users who require high-performance database administration and utility products. The LOADPLUS product is a flexible, high-performance tool that combines all of the basic load tasks with additional functionality.

**LOADPLUS benefits**

LOADPLUS offers you the following significant benefits:

- **Increases availability** of DB2 data because LOADPLUS reduces the time needed to load the data
  
  In addition, the online load options allow your tables to remain accessible in either read-only (RO) or read/write (RW) status while LOADPLUS loads your data. LOADPLUS also provides a unique, high-performance option that works with UNLOAD PLUS to reduce total processing time for unloading and reloading data.

- **Reduces costs** of loading DB2 data because LOADPLUS uses fewer CPU cycles and EXCPs

- **Provides flexibility** by allowing you to load data from a variety of input formats

- **Expands functionality** to eliminate the need to perform additional tasks before and after data is loaded
  
  For example, LOADPLUS provides the following functions:

  - **Comprehensive set of data type conversions** to reduce the need for special application code
  
  - **Ability to make image or inline copies** during the load
  
  - **Ability to dynamically allocate work files and copy data sets**, reducing the need to include DD statements in your JCL

- **Integrates with the BMC solutions for DB2**

**BMC Software solution integration**

LOADPLUS is also a component of the following solutions. For more information, see the release notes for these solutions.
BMC Object Administration for DB2 solution

The Object Administration for DB2 solution allows you to manage your DB2 databases. Using the solution simplifies the administration of your DB2 database, improves availability, and ensures data integrity.

This solution includes the following products and primary technology components:

- ALTER for DB2
- CATALOG MANAGER for DB2
- CHANGE MANAGER for DB2
- BMC Common Statistics
- COPY PLUS for DB2
- LOADPLUS for DB2
- SNAPSHOT UPGRADE FEATURE for DB2 (SUF)
- UNLOAD PLUS for DB2
- Workbench for DB2

BMC High Speed Utilities for DB2 solution

The High Speed Utilities for DB2 solution delivers high availability of objects during data and database maintenance. The solution also significantly reduces runtime costs by minimizing CPU consumption.

This solution includes the following products and primary technology components:

- CHECK PLUS for DB2
- BMC Common Statistics
- COPY PLUS for DB2
- LOADPLUS for DB2
- RECOVER PLUS for DB2
- REORG PLUS for DB2
- SNAPSHOT UPGRADE FEATURE for DB2
- UNLOAD PLUS for DB2

BMC Database Administration for DB2 solution

You can use the Database Administration solution to manage your DB2 databases quickly, efficiently, and effectively.

This solution includes the following products and primary technology components:

- CATALOG MANAGER for DB2
- CHANGE MANAGER for DB2
# Tasks that LOADPLUS performs

LOADPLUS accomplishes standard load tasks and also offers numerous functional enhancements in the following areas:

## Table 1: LOADPLUS functional enhancements

<table>
<thead>
<tr>
<th>Functional area</th>
<th>Enhanced tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Resources</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Analyzes and allocates memory and CPU resources for maximum throughput</td>
</tr>
<tr>
<td></td>
<td>■ Optionally analyzes data set resources that are needed for the specific load job</td>
</tr>
<tr>
<td></td>
<td>■ Optionally offloads eligible processing to a zIIP</td>
</tr>
<tr>
<td></td>
<td>■ Optionally estimates input file size automatically for dynamic allocation and efficient sort processing</td>
</tr>
<tr>
<td></td>
<td>■ Optionally allocates input, load data, index work, sort work, discard, error, and copy files dynamically</td>
</tr>
<tr>
<td><strong>Indexes</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Optionally reorganizes the participating indexes</td>
</tr>
<tr>
<td></td>
<td>■ Optionally updates indexes, which can improve performance when loading a small number of rows into an existing table that has a large number of rows</td>
</tr>
<tr>
<td></td>
<td>■ Optionally skips building or updating nonpartitioned indexes</td>
</tr>
<tr>
<td>Functional area</td>
<td>Enhanced tasks</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Performance</td>
<td>■ Optionally provides single-phase load processing, which combines the PRELOAD and LOAD phases into one phase (the COMBINED phase)</td>
</tr>
<tr>
<td></td>
<td>■ Provides an option to order data by table and clustering key or by clustering key alone</td>
</tr>
<tr>
<td></td>
<td>■ Optionally defers checks for duplicate keys in unique indexes</td>
</tr>
<tr>
<td></td>
<td>■ Provides an option to improve performance by not sorting the data or clustering index if the data is already in correct clustering index sequence</td>
</tr>
<tr>
<td></td>
<td>■ Provides an option to improve performance when transferring data between similarly defined tables by loading data that was unloaded by UNLOAD PLUS in an internal format</td>
</tr>
<tr>
<td>Object availability</td>
<td>■ Provides an option for the utility to pause in its processing before the data is loaded if input records are discarded</td>
</tr>
<tr>
<td></td>
<td>■ Runs concurrently with other BMC utilities on DB2 spaces</td>
</tr>
<tr>
<td></td>
<td>■ Provides an option to leave tables in RO status while loading your data</td>
</tr>
<tr>
<td></td>
<td>■ Provides an option to leave tables in RW status while loading your data</td>
</tr>
<tr>
<td>Compression</td>
<td>Builds or keeps a compression dictionary when the table space or partition being loaded is defined with the COMPRESS YES attribute</td>
</tr>
</tbody>
</table>
### Functional area

**Loading data**

- Loads data from one or more data sets into one or more tables or table space partitions in a single command execution, either replacing or adding to the existing data
- Optionally sorts the input data by the clustering key or partitioning key
- Optionally orders rows by table
- Allows you to load data that was archived by the REORG PLUS product
- Allows you to load data that was unloaded by the UNLOAD PLUS product in an internal format
- Allows you to load data from batch pipes
- Provides powerful selection criteria for specifying which input records to load
- As part of the load, optionally deletes and redefines user-defined data sets and data sets that are defined in DB2 storage groups
- Avoids adding rows and keys to the table spaces and indexes if they cause duplicate keys in a unique index, so that the rows and keys do not need to be deleted later in the LOAD phase
  
  Following this process, your table spaces and indexes remain organized.

### Data integrity

- Optionally checks the following constraints:
  
  - DB2-defined check constraints
  - DB2-defined referential constraints
  - Informational referential constraints

### Copies

- With a self-contained copy function, produces image copies or DSN1COPY-type copies concurrently with the load
- Optionally dynamically allocates the image copy data sets
- Optionally produces inline image copies *while* loading your tables, rather than DSN1COPY-type copies or image copies *after* loading your tables
<table>
<thead>
<tr>
<th>Functional area</th>
<th>Enhanced tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statistics and reports</td>
<td>■ Optionally updates statistics in the DB2 catalog so that the DB2 optimizer can use them</td>
</tr>
<tr>
<td></td>
<td>■ Optionally updates statistics in the DASD MANAGER PLUS database statistics tables</td>
</tr>
<tr>
<td></td>
<td>■ Produces a statistics report in the SYSPRINT output or, optionally, in a separate data set</td>
</tr>
<tr>
<td></td>
<td>■ Updates the DB2 real-time statistics tables</td>
</tr>
<tr>
<td></td>
<td>■ Prevents the output of unnecessary discard messages and records by allowing you to ignore specified types of discards</td>
</tr>
<tr>
<td></td>
<td>■ Reports multiple load command errors at one time, reducing the number of runs needed to correct your load specification</td>
</tr>
</tbody>
</table>
### Functional area

**Enhanced tasks**

- Provides a comprehensive set of data type conversions and support for user-written exit routines that perform special data conversions
- Provides options to perform data translation
- Optionally provides a PRELOAD phase for verifying that your input data is correct before your existing data is replaced (and lost), without incurring the overhead of actually loading the data
- Provides full Boolean logic on the WHEN condition (allows AND, OR, NOT, and parentheses)
- Provides control over expressions used in WHEN, NULLIF, and DEFAULTIF processing through the RULES installation option
- Allows you to concatenate separate physical input records into one larger logical record
- Allows you to assign a constant value or CURRENT DATE, CURRENT TIME, or CURRENT TIMESTAMP to a column
- Allows multiple NULLIF and DEFAULTIF conditions on a field specification
- Provides powerful criteria for assigning NULL or default values to columns:
  - Allows you to assign default values to nullable or nondefaultable columns
  - Allows you to specify the value for a DEFAULTIF value, overriding the standard default
  - Allows you to assign NULL or a default value to a column if a conversion error occurs while LOADPLUS is processing the column’s input value

### Job and object status

- Provides restart capabilities
- Optionally allows you to set user-defined return codes for a range of situations
- Sets the status of all affected table spaces to CHECK pending (CHKP) if referential integrity constraints or check constraints exist and were not checked
Differences between LOADPLUS and the IBM DB2 LOAD utility

This topic describes LOADPLUS functions and operations that differ from functions and operations in the IBM DB2 LOAD utility.

**Note**

When LOADPLUS invokes DSNUTILB, you are using the IBM DB2 LOAD utility. Therefore, many of these differences do not apply for a DSNUTILB load.

<table>
<thead>
<tr>
<th>Functional or operational area</th>
<th>Differences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization</td>
<td>LOADPLUS does not run as part of the DB2 subsystem. Therefore, you must have system authorization that is similar to that required by DB2 unless you are using the IBM Resource Access Control Facility (RACF) component of the z/OS Security Server and LOADPLUS is installed with OPNDB2ID=YES.</td>
</tr>
<tr>
<td>Object status</td>
<td>For SHRLEVEL NONE, the table space that you are loading is stopped at the beginning of the PRELOAD or LOAD phase, or during the COMBINED phase. LOADPLUS starts the table space at the end of the LOAD or COMBINED phase.</td>
</tr>
</tbody>
</table>
| Compression                    | - If the object that you are loading has the COMPRESS YES attribute, LOADPLUS always reserves enough pages to hold the largest compression dictionary.  
  - LOADPLUS optionally compresses XML data. |
| Tables                         | For segmented table spaces, you can specify LOAD RESUME YES INTO TABLE tableName REPLACE to load only the tables in which you want to replace data, leaving the other tables intact. |
| Indexes                        | - Unless you specify INDEX UPDATE or SKIPIX, LOADPLUS reorganizes all indexes that participate in a LOAD RESUME YES job by unloading existing index entries, merging in new index entries, and then sorting and reloading them into the index spaces. If you specify PART n REPLACE or TABLE tableName REPLACE, LOADPLUS rebuilds the data-sorting indexes for the partitions or tables that are being replaced.  
  - For partitioned table spaces, you can tell LOADPLUS to skip the index building process for nonpartitioned indexes that are associated with the table that you are loading. LOADPLUS supports this functionality for all SHRLEVEL values except SHRLEVEL CHANGE SQLAPPLY. |
### Differences between LOADPLUS and the IBM DB2 LOAD utility

<table>
<thead>
<tr>
<th>Functional or operational area</th>
<th>Differences</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFINE NO objects</td>
<td>To enable loading data into tables whose table spaces or index spaces were defined with DEFINE NO, LOADPLUS can materialize the underlying data sets without actually loading data. LOADPLUS materializes the data sets by performing a DB2 INSERT and ROLLBACK on the table. <strong>Note:</strong> For an SQLAPPLY load, DEFINE NO data sets are materialized by the DB2 INSERT processing that occurs during the COMBINED phase.</td>
</tr>
<tr>
<td><strong>Identity columns</strong></td>
<td>• LOADPLUS allows you to set the size of the cache that LOADPLUS uses when loading generated identity column values.</td>
</tr>
<tr>
<td></td>
<td>• LOADPLUS allows you to specify whether you want to reset the value of MAXASSIGNEDVAL in the SYSIBM.SYSSEQUENCES table when loading identity column values from an input file. When generating identity column values, LOADPLUS always updates MAXASSIGNEDVAL.</td>
</tr>
<tr>
<td></td>
<td>• For most load types, you can tell LOADPLUS whether to discard or fail when a generated identity column value would fall outside the range that is defined on the column.</td>
</tr>
<tr>
<td></td>
<td>• Unless you tell LOADPLUS not to enforce check constraints, LOADPLUS fails when attempting to generate identity column values if the identity column is referenced in the check condition of a DB2 check constraint.</td>
</tr>
<tr>
<td><strong>Availability</strong></td>
<td>• LOADPLUS allows you to specify SHRLEVEL REFERENCE with a LOAD REPLACE or LOAD RESUME YES job. SHRLEVEL REFERENCE allows RO access to the table space that you are loading.</td>
</tr>
<tr>
<td></td>
<td>• LOADPLUS allows you to specify SHRLEVEL CHANGE with a LOAD REPLACE job. SHRLEVEL CHANGE allows RW access to the table space that you are loading.</td>
</tr>
<tr>
<td>Functional or operational area</td>
<td>Differences</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>LOAD command options</td>
<td>■ LOADPLUS allows only one LOAD command in the command input data set (SYSIN).</td>
</tr>
<tr>
<td></td>
<td>■ LOADPLUS optimizes the processing of INTO PART statements by combining statements that are the same. For example, if you specify INTO PART 1 twice on the same LOAD command, LOADPLUS combines these statements into a single INTO statement and loads one set of rows. In contrast, the DB2 LOAD utility loads two sets of rows into partition 1 in this example.</td>
</tr>
<tr>
<td></td>
<td>■ For a field specification, LOADPLUS requires that you specify the position before the data type in the command statement.</td>
</tr>
<tr>
<td>WHEN, NULLIF, and DEFAULTIF processing</td>
<td>LOADPLUS provides the RULES installation option for specifying the set of comparison rules to be used in WHEN, NULLIF, and DEFAULTIF processing. Specifying RULES=STANDARD causes comparisons to be performed directly on the input record values rather than the values as they will be in the column. Specifying RULES=BMC causes comparisons to be performed on the values as they will be in the column.</td>
</tr>
<tr>
<td>Constraints and referential integrity</td>
<td>■ LOADPLUS optionally allows you to enforce informational referential constraints.</td>
</tr>
<tr>
<td></td>
<td>■ LOADPLUS does not support referential checking on self-referencing tables.</td>
</tr>
<tr>
<td></td>
<td>■ LOADPLUS does not support referential checking when loading the parent and child in the same load execution.</td>
</tr>
<tr>
<td></td>
<td>■ If check constraint conditions contain one or more of the following constant types, LOADPLUS does not enforce those check constraints:</td>
</tr>
<tr>
<td></td>
<td>— Floating-point constants</td>
</tr>
<tr>
<td></td>
<td>— Constants whose length is greater than 254 bytes</td>
</tr>
<tr>
<td>Data types and conversions</td>
<td>■ LOADPLUS always truncates digits for conversions of FLOAT to DECIMAL if digits are not needed in the precision. This action differs from the DB2 LOAD utility action of rounding in certain circumstances, but is consistent with the LOADPLUS definition of numeric-to-numeric conversions. If rounding is required, use the ROUND keyword on your field specification.</td>
</tr>
<tr>
<td></td>
<td>■ LOADPLUS supports all floating-point formats except IEEE Binary Floating Point.</td>
</tr>
<tr>
<td>Functional or operational area</td>
<td>Differences</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Multiple data sets and multitasking</td>
<td>LOADPLUS automatically multitasks when you specify multiple data sets. You can specify multiple SYSREC input data sets and use multiple SORTOUT data sets to load partitioned tables. You can also specify multiple SORTWK data sets to multitask sort processing and multiple SYSUT1 data sets to multitask index processing.</td>
</tr>
<tr>
<td>Work data sets</td>
<td>▪ When allocating SORTOUT data sets, LOADPLUS requires that the total allocation be large enough to contain the DB2 row images that are created during the PRELOAD or COMBINED phase.</td>
</tr>
<tr>
<td></td>
<td>▪ For most single-phase load jobs, LOADPLUS does not use the SORTOUT and SYSUT1 work data sets and does not require that you define them in your JCL. For some single-phase load jobs, however, these data sets are required if you want to be able to restart the job. For details about when these data sets are required for restart, see “SORTOUT data sets in LOADPLUS” on page 443 and “SYSUT1 data sets in LOADPLUS” on page 459.</td>
</tr>
<tr>
<td>Copies</td>
<td>▪ You can create one or more image copies by using the LOADPLUS COPY YES option, regardless of whether you specify LOAD REPLACE, LOAD RESUME NO, or LOAD RESUME YES. You cannot create an image copy when performing an SQLAPPLY load.</td>
</tr>
<tr>
<td></td>
<td>▪ LOADPLUS creates all image copies with system pages at the beginning of the data set (in the same way that the IBM COPY utility creates image copies when you specify SYSTEMPAGES YES). However, LOADPLUS does not provide a SYSTEMPAGES option.</td>
</tr>
<tr>
<td></td>
<td>▪ LOADPLUS optionally creates an inline copy of a LOB table space.</td>
</tr>
<tr>
<td>Free space</td>
<td>When loading XML data, LOADPLUS applies FREEPAGE specifications to the XML table space and the base table.</td>
</tr>
<tr>
<td>Functional or operational area</td>
<td>Differences</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Recoverability and restartability</td>
<td>- LOADPLUS uses work files until it loads all of the data. Therefore, the work files should not be temporary data sets if you want to be able to restart the job.</td>
</tr>
<tr>
<td></td>
<td>- LOADPLUS allows an image copy to be made after a table is loaded even if LOADPLUS leaves the table in CHECK pending status. Because the CHECK DATA utility logs any changes made to the table, data recovery is possible using the RECOVER utility, which uses the full image copy and applies log records as necessary, ensuring the integrity of the data in the table. A RECOVER TOCOPY utility execution places the affected table in CHECK pending status again. A subsequent invocation of the CHECK DATA utility restores referential integrity.</td>
</tr>
<tr>
<td></td>
<td>- For all types of load jobs except an SQLAPPLY load, LOADPLUS does not write to the DB2 log, even when your table space is defined as LOGGED. Therefore, you must create a full image copy to ensure recoverability of the table space after loading. You can create a full image copy by using the LOADPLUS COPY YES option. You can also use COPY PLUS or the IBM DB2 COPY utility. An SQLAPPLY load does write to the DB2 log, unless your table space is defined as NOT LOGGED. In addition, you cannot create an image copy with this type of load in LOADPLUS.</td>
</tr>
<tr>
<td>COPY pending status</td>
<td>- When you specify COPY YES, LOADPLUS resets COPY pending status after loading the data and starting the table space.</td>
</tr>
<tr>
<td></td>
<td>- When you specify COPY NO, LOADPLUS places the table space in COPY pending status only if the value of COPYPEND is YES. If the value of COPYPEND is NO, LOADPLUS does not place the table space in COPY pending status.</td>
</tr>
<tr>
<td>SYSMAP data set</td>
<td>LOADPLUS uses a SYSMAP data set only when invoking DSNUTILB.</td>
</tr>
<tr>
<td>Multiple volumes</td>
<td>For multi-volume STOGROUP-defined table spaces and indexes, LOADPLUS attempts to reallocate the data set on the volume on which it currently resides if the volume is still defined in the storage group. After the current volume, the order of the volumes retrieved from the storage group is unpredictable.</td>
</tr>
<tr>
<td>Concurrent utilities</td>
<td>LOADPLUS uses the BMCSYNC table to control access to DB2 spaces by BMC utilities. This allows LOADPLUS and other BMC utilities to run concurrently on DB2 spaces.</td>
</tr>
<tr>
<td>Return codes</td>
<td>When the load job discards records (and no other conditions or options result in a greater return code), LOADPLUS completes with return code 0.</td>
</tr>
</tbody>
</table>
How LOADPLUS works

This topic describes the LOADPLUS processing phases and the data sets and common components that LOADPLUS uses.

LOADPLUS processing phases

This topic describes the processing phases of LOADPLUS, their primary functions, and the data sets that are associated with each phase.

The following table describes the processing phases of LOADPLUS and their primary functions:

Table 3: LOADPLUS processing phases

<table>
<thead>
<tr>
<th>Phase</th>
<th>Description</th>
</tr>
</thead>
</table>
| UTILINIT | During this phase, LOADPLUS completes the following processes:  
  ■ Initializes the job  
  ■ Reads, parses, and verifies the LOAD command  
  ■ Performs DB2 catalog lookup  
  ■ If you enabled automated file size estimation, estimates input file size  
  ■ If any referenced object is VCAT-defined, verifies the commands in the SYSIDCIN data set |
| DSNUTILB | LOADPLUS executes this phase only when the type of load that you are running requires it. During this phase, LOADPLUS passes processing to DSNUTILB. This phase takes the place of the LOADPLUS PRELOAD, LOAD, and COMBINED phases. Additionally, this phase performs some of the tasks that the UTILTERM phase normally performs, such as updating statistics. For more information about the types of load jobs that require this phase, see “Load jobs that invoke DSNUTILB” on page 57. |
| ANALYZE | During this optional phase, LOADPLUS completes the following processes:  
  ■ Analyzes the objects that are being loaded  
  ■ Optionally produces statistics to help determine data set sizes for allocating those data sets  
LOADPLUS executes this phase when dynamic data set allocation is active or when you specify the ANALYZE command option. LOADPLUS uses information from this phase to optimize the load process. |
<table>
<thead>
<tr>
<th>Phase</th>
<th>Description</th>
</tr>
</thead>
</table>
| PRELOAD   | During this phase, LOADPLUS reads data from the input data sets. Additionally, LOADPLUS completes the following processes:  
  ■ Converts data as needed and verifies that the data is correct  
  ■ Builds DB2 rows in the SORTOUT data set  
  ■ Builds index information in the SYSUT1 data set  
  ■ Writes input records that are in error to the SYSDISC discard data set  
  ■ Builds the compression dictionary and compresses the data rows  
  ■ Invokes the BMCSORT component when applicable  
  ■ Reads the index and table space of the table or partition being loaded, if you are loading rows into tables or partitions that already contain data  
  The SYSREC and SYSDISC data sets are freed at the end of this phase and you should not specify FREE=CLOSE for these data sets. |
| LOAD      | During this phase, LOADPLUS completes the following processes:  
  ■ Redefines the VSAM data sets if the value of the REDEFINE option is YES  
  ■ Reads the rows from the SORTOUT and SYSUT1 data sets  
  ■ Loads the data into the table space and indexes  
  ■ Adds partitions for partition-by-growth table spaces if required (SHRLEVEL NONE)  
  ■ Invokes BMCSORT for indexes that are not sorted in the PRELOAD phase  
  ■ Builds or updates indexes  
  ■ Creates any requested copies and registers them with the DB2 catalog |
<p>| COMBINED  | This phase combines all functions of the PRELOAD and LOAD phases into a single processing phase. In most cases, the COMBINED phase avoids the intermediate steps of writing to the SORTOUT and SYSUT1 data sets; instead, LOADPLUS either writes data directly to the table space and index space, or sends data directly to the High-speed Apply Engine to be inserted into the table space and index space. |</p>
<table>
<thead>
<tr>
<th>Phase</th>
<th>Description</th>
</tr>
</thead>
</table>
| UTILTERM   | During this phase, LOADPLUS completes the following cleanup tasks:  
■ Sets the pending status of all affected tables and table spaces  
■ Updates the BMCHIST table  
■ Updates the DASD MANAGER PLUS database statistics tables  
■ Updates the DB2 catalog statistics  
■ Updates the DB2 real-time statistics tables  
■ Adds partitions for partition-by-growth table spaces if required (SHRLEVEL REFERENCE or SHRLEVEL CHANGE)  
■ Performs the FASTSWITCH or rename operations that are associated with the staging data sets for SHRLEVEL CHANGE (except an SQLAPPLY load) and SHRLEVEL REFERENCE  
■ Performs a clone exchange, if requested  
■ Cleans up the BMCSYNC and BMCUTIL tables  
LOADPLUS executes only part of this phase when invoking DSNUTILB. |
| all phases | During all phases except the DSNUTILB phase, LOADPLUS updates the BMCUTIL and BMCSync DB2 tables.                                                                                                              |

The following figures illustrate the possible input and output data sets that are associated with each phase of processing. These figures also show, in some cases, where LOADPLUS calls common components.
Figure 1: Processing phases of a two-phase load

SYSIN → UTILINIT phase → DB2 catalog

Indexes → ANALYZE phase → SYSIDCIN

SYSRECnn → PRELOAD phase → SORTWKnn

SYSDISC → LOAD phase → SORTWKnn

SORTOUTn → LOAD phase

SYSERR → LOAD phase

SYSUT1nn → LOAD phase

Copy data sets → LOAD phase

Table spaces or staging table space → UTILTERM phase

Indexes or staging indexes → UTILTERM phase

DASD MANAGER PLUS database → UTILTERM phase → DB2 catalog

All phases use the SYSPRINT data set.

All sort processing uses the UTPRINT data set.
How LOADPLUS works

Figure 2: Processing phases of a single-phase load

All phases use the SYSPRINT data set.

All sort processing uses the UTPRINT data set.
All phases use the SYSPRINT data set. All sort processing uses the UTPRINT data set.
Figure 4: Processing phases when invoking DSNUTILB

The UTILINIT and UTILTERM phases use the SYSPRINT data set.
LOADPLUS data sets

This topic briefly describes the data sets that LOADPLUS uses.

You can override most of the ddnames or ddname prefixes by using LOAD command options. For more information about how to specify the DD statement names and data set allocation sizes, and when the data sets are required, see “LOADPLUS DD statements” on page 432.

Table 4: LOADPLUS data sets

<table>
<thead>
<tr>
<th>Data set or ddname</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>APTDUMP</td>
<td>APTDUMP is the data set that contains SNAP dump output for an SQLAPPLY load.</td>
<td>“APTDUMP data sets in LOADPLUS” on page 435</td>
</tr>
<tr>
<td>ASUSRPRT</td>
<td>ASUSRPRT is an optional data set that contains the statistics report that is generated when you specify BMCSTATS YES, BMCSTATS REPORT, or UPDATEDB2STATS YES.</td>
<td>“ASUSRPRT data sets” on page 436</td>
</tr>
<tr>
<td>BMCFORCE</td>
<td>BMCFORCE is an optional data set that contains the thread cancelation report that is generated when you specify either FORCE REPORTONLY or FORCE_RPT YES.</td>
<td>“BMCFORCE data sets” on page 436</td>
</tr>
<tr>
<td>Copy data sets</td>
<td>LOADPLUS can create one or more output copy data sets. These output data sets contain a DSN1COPY-type copy or image copy. After completing the copy, LOADPLUS deallocates the data set to free the device unless you specified VOL=(,RETAIN) in your JCL.</td>
<td>“Copy data sets in LOADPLUS” on page 436</td>
</tr>
<tr>
<td>DB2 data sets</td>
<td>LOADPLUS dynamically allocates the DB2 data sets (table spaces or index spaces) that you are loading. Therefore, you do not need to specify them in your JCL.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>SORTOUTn</td>
<td>For a two-phase load, SORTOUTn is the data set that contains the output of the PRELOAD phase. This data set becomes the input data set to the LOAD phase. The n is required only if you specify multiple data sets. LOADPLUS uses the data set during the PRELOAD phase through the end of the LOAD phase. After the table space is loaded, LOADPLUS deallocates the data set to free the device. You can also use this data set with certain single-phase load jobs if you want to be able to restart the job.</td>
<td>“SORTOUT data sets in LOADPLUS” on page 443</td>
</tr>
</tbody>
</table>
### Data set or ddname | Description | Reference
--- | --- | ---
**SORTWKnn**<sup>a</sup> | SORTWK is the work data set that BMCSORT uses. This data set is used in the PRELOAD and LOAD phases for a two-phase load, and in the COMBINED phase for a single-phase load. You cannot allocate the sort work files as VIO data sets or tape data sets. You must allocate each individual work file on a single DASD unit. | “SORTWK data sets in LOADPLUS” on page 446
**SYSDISC**<sup>a</sup> | SYSDISC is the data set that contains the discarded input records. LOADPLUS uses this data set in the PRELOAD phase for two-phase load and the COMBINED phase for single-phase processing. At the end of the PRELOAD or COMBINED phase, LOADPLUS deallocates the data set. | “SYSDISC data set in LOADPLUS” on page 450
**SYSERR**<sup>a</sup> | SYSERR is the data set that contains information about errors. LOADPLUS uses this data set in the PRELOAD phase through the end of the LOAD phase for two-phase load, and throughout the COMBINED phase for single-phase processing. | “SYSERR data set in LOADPLUS” on page 453
**SYSIDCIN** | SYSIDCIN is the input data set that contains the IDCAMS command statements that LOADPLUS uses to redefine user-defined (VCAT-defined) data sets. These statements include statements to redefine the VCAT-defined staging data sets that LOADPLUS uses when running a load with SHRLEVEL CHANGE (except an SQLAPPLY load) or SHRLEVEL REFERENCE. LOADPLUS reads, parses, and partially verifies the SYSIDCIN data set in the UTILINIT phase. LOADPLUS issues the IDCAMS commands before loading the VCAT-defined data sets. | “SYSIDCIN data set in LOADPLUS” on page 462
**SYSIN** | SYSIN is the input data set that contains the LOAD command. | “SYSIN data sets” on page 454
**SYSMAP** | SYSMAP is the additional data set that is required when invoking DSNUTILB from LOADPLUS. This data set contains mapping information between the table row identifier and the input record. | “SYSMAP data set in LOADPLUS” on page 454
**SYSPRIN2** | The SYSPRIN2 output data set contains the same LOADPLUS messages that are output to SYSPRINT. In a worklist environment, SYSPRIN2 enables you to view LOADPLUS output in real time. | “SYSPRIN2 data sets” on page 455
**SYSPRINT** | SYSPRINT is the output data set that contains LOADPLUS messages. All phases use this data set. | “SYSPRINT data sets” on page 455
### Associated products and common components that LOADPLUS uses

In addition to its own processing components, LOADPLUS uses the BMC products and common components that this topic describes.

#### Table 5: Products and components that LOADPLUS uses

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC Common Statistics</td>
<td>The BMC Common Statistics component of DASD MANAGER PLUS provides a common method for updating DB2 catalog statistics and DASD MANAGER PLUS statistics tables. This component also provides statistics reporting. This component is installed during LOADPLUS installation, but is maintained separately from LOADPLUS.</td>
</tr>
<tr>
<td>BMCSORT</td>
<td>The BMCSORT technology is a common BMC technology. LOADPLUS uses BMCSORT to allocate sort work files and perform sort processing. This component is installed during LOADPLUS installation, but is maintained separately from LOADPLUS.</td>
</tr>
<tr>
<td>Component</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>DB2 Solution Common Code</strong></td>
<td>BMC DB2 Solution Common Code (SCC) is a set of technologies that provide common processes for several BMC products for DB2. LOADPLUS uses SCC technologies for such processes as setting object statuses and compressing data. This component is installed during LOADPLUS installation, but is maintained separately from LOADPLUS.</td>
</tr>
<tr>
<td><strong>DB2 Utilities Common Code (D2U)</strong></td>
<td>D2U is a set of technologies that provides common processes for the BMC Utility products for DB2 and the DASD MANAGER PLUS product. This component is installed during LOADPLUS installation, but is maintained separately from LOADPLUS.</td>
</tr>
<tr>
<td><strong>High-speed Apply Engine</strong></td>
<td>The High-speed Apply Engine is a component of the BMC Log Master for DB2 product. LOADPLUS uses High-speed Apply to load data when performing an SQLAPPLY load. LOADPLUS passes processing to High-speed Apply during the COMBINED phase. High-speed Apply loads the data from LOADPLUS by performing SQL INSERT processing. This component is installed during LOADPLUS installation, but is maintained separately from LOADPLUS.</td>
</tr>
</tbody>
</table>
| **EXTENDED BUFFER MANAGER (XBM) or SNAPSHOT UPGRADE FEATURE (SUF)** | XBM or SUF provides the following capabilities:  
  ■ zIIP processing  
  ■ For LOAD RESUME YES SHRLEVEL REFERENCE, a snapshot image of data in a table space  
  ■ During LOADPLUS referential constraint checking, a consistent image of the primary index on the parent table of the table that you are loading, while keeping the index available  
XBM and SUF are licensed, installed, and maintained separately from LOADPLUS. |
Operational considerations in LOADPLUS

This chapter provides procedures and information to consider when using LOADPLUS.

System setup for LOADPLUS

Review the following topics for recommendations and requirements before you use the LOADPLUS for DB2 product.

DB2 support in LOADPLUS

This version of LOADPLUS supports IBM DB2 Versions 10 (new-function mode only) and 11.

LOADPLUS system and software requirements

The LOADPLUS product has the following requirements:

- This version of LOADPLUS has the following system requirements:
  - IBM System z10 processor or a later 64-bit z/Architecture processor
  - z/OS Version 1.10 or later

- For all types of load jobs, you must have a minimum of the following versions of BMC common components:
  - Version 2.4.01 of BMCSORT
  - Version 11.2.00 of the DB2 Utilities Common Code (D2U)
  - Version 11.1.00 with PTF BPJ0944 of the DB2 Solution Common Code (SCC)
To update DB2 catalog statistics or the DASD MANAGER PLUS database statistics tables, you must have installed a minimum of version 11.2.00 of the BMC Common Statistics component.

To perform a LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY load, you must have installed a minimum of version 11.1.00 of the BMC High-speed Apply Engine product.

For the following features, LOADPLUS requires version 6.1.00 or later of either the BMC EXTENDED BUFFER MANAGER (XBM) product or its SNAPSHOT UPGRADE FEATURE (SUF) technology.

— When checking referential constraints during your load job

— When you specify LOAD RESUME YES SHRLEVEL REFERENCE

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 snapshot processing is required, LOADPLUS searches for an XBM subsystem at this level.

To offload eligible processing to a zIIP, LOADPLUS requires version 6.1.00 or later 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, LOADPLUS searches for an XBM subsystem at this level.

To use any features that invoke DSNUTILB, you must be licensed to use the IBM DB2 LOAD utility.

**Note**

For information about the features that invoke DSNUTILB, see the *LOADPLUS for DB2 Reference Manual*.

---

**Required authorizations for LOADPLUS**

Using the LOADPLUS product requires that you have the appropriate authorization within DB2 and through your system security package, such as the Resource Access Control Facility (RACF).

You need sufficient authorization to access resources and perform the tasks that are accomplished during LOADPLUS processing, as described in this topic.
Authorization verification mechanisms for the Utility products

The products use one of the following mechanisms to verify authorization:

- If the DB2 DSNX@XAC authorization exit is available for your system, the product 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 the CA-ACF2 and CA-Top Secret products.

  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, the product uses the standard DB2 method to check security.

Setting LOADPLUS authorizations

LOADPLUS does not run as part of the DB2 subsystem. Therefore, users must have system and data set authorizations that are equivalent to the authorizations that DB2 requires. Use the following procedures to set the necessary authorizations.

To set DB2 authorizations

1. For all load jobs, grant the following authorizations:
   - Sufficient DB2 authority to execute the LOADPLUS plan and all packages that the LOADPLUS plan uses
   - Authorization equivalent to the authorization that the IBM DB2 LOAD utility requires

2. To enable loading tables that contain identity columns, also complete the following authorization steps:

   Note
   These additional authorizations might be implicit in the authorization that the users have.

   a. Grant SELECT privileges on the following DB2 tables:
To enable use of the UPDatemaxa YES option to update the MAXASSIGNEDVAL column of the SYSIBM.SYSEQUENCES table, complete the following steps:

1. Determine which of the following authorization IDs should have ALTER privileges for the table that is being loaded:
   - User ID of the job owner
   - INSTALL SYSADM

2. Ensure that the value for the UPDatemaxa_AUTHID installation option reflects this determination.

3. Grant ALTER privileges on the table that is being loaded for the appropriate authorization ID.

To enable loading a table whose table space or index spaces are defined with DEFINE NO, also grant INSERT privileges on that table.

**Note**

INSERT privileges might be implicit in the authority that the users have.

To enable the use of the FORCE option to cancel DB2 threads that might prevent a successful drain during a load job, also grant the following authorizations:

- DISPLAY privileges
- One of the following authorities:
  - SYSADM
  - SYSOPR
  - SYSCTRL

**Note**

These authorizations might be implicit in the authority that the users have.

To enable zIIP processing and LOADPLUS features that use snapshot processing, ensure that you have the appropriate authorizations for XBM or SUF.

For information about security levels and authorizations for XBM, see the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide.

To enable running an SQLAPPLY load, also grant the following authorizations:
When running an SQLAPPLY load, LOADPLUS passes processing during the COMBINED phase to the High-speed Apply Engine component of the BMC Log Master for DB2 product. High-speed Apply requires the following DB2 authorizations. The APTGRANT member of the High-speed Apply HLQ.LLQSAMP installation data set (where HLQ is the high-level qualifier that is set during installation and LLQ is the low-level qualifier or prefix set during installation) contains sample authorization statements.

You can use secondary authorization IDs to limit access as necessary for your site.

- **(Normally granted during High-speed Apply installation) **EXECUTE privileges:
  - EXECUTE privilege for the plan that High-speed Apply uses to access its own restart table and the catalog
  - EXECUTE privilege for the High-speed Apply restart package

- **(Normally granted after High-speed Apply installation) **additional privileges:
  - INSERT privileges on the table that a user is loading
  - INSERT, UPDATE, SELECT, and DELETE privileges on the High-speed Apply restart table
  - CREATE privileges for the collections that High-speed Apply creates
  - Bind privileges with the add option (BINDADD) for the plans and packages that High-speed Apply creates during apply processing

The High-speed Apply Engine provides several ways to grant the CREATE and BINDADD privileges. Some techniques avoid granting bind privileges to the user ID that runs High-speed Apply. For more information, see the High-speed Apply Engine Reference Manual.

---

**Note**

The pre-bound plan option, described in the High-speed Apply Engine Reference Manual, is not compatible with LOADPLUS.

---

**To enable data set access using the DB2 RACF ID**

1 Specify OPNDB2ID=YES in your installation options.

This option tells LOADPLUS to use the DB2 RACF ID for data set access.
To enable data set access when not using the DB2 RACF ID

1 Specify OPNDB2ID=NO in your installation options.

This option tells LOADPLUS not to use the DB2 RACF ID for data set access.

2 If using RACF or a similar system security package to protect underlying data sets and the Integrated Catalog Facility (ICF) catalog of a table or index space, set a minimum of the levels of authorization shown in the following table for all load jobs.

Table 6: Minimum levels of authorization that LOADPLUS requires

<table>
<thead>
<tr>
<th>Table or index space definition</th>
<th>To access, update, and define DB2 data sets</th>
<th>To access and update the ICF catalog</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCAT-defined</td>
<td>CONTROL</td>
<td>UPDATE</td>
</tr>
<tr>
<td>STOGROUP-defined</td>
<td>ALTER or CONTROL</td>
<td>UPDATE or CONTROL</td>
</tr>
</tbody>
</table>

The following steps illustrate one method for granting these data set authorizations when your site uses a system security package other than RACF:

1 Associate users with a security group.

2 Grant EXECUTE privileges on the LOADPLUS product program (AMUUMAIN) to the security group.

3 Grant the data set authorizations to AMUUMAIN.

3 To enable checking referential constraints during the load, also grant READ privileges on the primary index of the parent table for the table being loaded.

4 To enable using rename or FASTSWITCH processing, if you establish authority at a node lower than the highest node, grant the same privileges as shown in Table 6 on page 54 for the following data sets:

- When FASTSWITCH NO is in effect:
  - VCAT.BMCDBD.database.object.I0001
  - VCAT.BMCDBC.database.object.I0001
  - VCAT.OLDDBD.database.object.I0001
  - VCAT.OLDDBC.database.object.I0001
  - VCAT.BMCDBD.database.object.J0001
  - VCAT.BMCDBC.database.object.J0001
  - VCAT.OLDDBD.database.object.J0001
  - VCAT.OLDDBC.database.object.J0001
■ When FASTSWITCH YES is in effect:
  — VCAT.BMCDBD.database.object.I0001
  — VCAT.BMCDBC.database.object.I0001
  — VCAT.BMCDBD.database.object.J0001
  — VCAT.BMCDBC.database.object.J0001

Overriding the MEMLIMIT system parameter

The utility requires above-the-bar memory and might abend if sufficient memory is not available.

The default value for the System Management Facility (SMF) MEMLIMIT parameter is 2 GB. This value is set in member SMFPRMxx in SYS1.PARMLIB.

**To override the default MEMLIMIT value**

1. Use one of the following methods to override the default MEMLIMIT value:
   - *(recommended)* Specify REGION=0M in the JCL.
   - Specify the one of the following values for the MEMLIMIT parameter in the JCL, or in an SMF IEFUSI exit:
     - Specify NOLIMIT to allow unlimited above-the-bar memory.
     - If you are unable to specify NOLIMIT, specify at least 4 GB. If you are operating on LOB or XML data, specify at least 32 GB.

Customizing products that prevent x37 abends in LOADPLUS

Products that prevent x37 abends must be customized to ensure that they work properly with EXCP processing in LOADPLUS.

When inadequate space is available for work data sets during job execution, the system issues an x37 abend and ends the job. Some sites use products such as the BMC MainView Storage Resource Manager (SRM) StopX37/II product to allocate additional volumes automatically when this condition occurs. However, those products might fail to intercept x37 abends if EXCP processing is in use.

LOADPLUS uses EXCP processing. Complete the following procedure to ensure proper handling of x37 abends.
To prevent x37 abends in LOADPLUS

1. Determine whether your site uses a product that intercepts x37 abends and whether that product is sensitive to EXCP processing.

See your DASD storage management system administrator for assistance.

2. If you use MainView SRM StopX37, use one of the following methods to customize the product to prevent x37 abends in LOADPLUS.

   **Note**
   If you use a similar product from another vendor, see that product’s documentation regarding activating support for EXCP processing.

   - Update the System Master Global member (the active SMMSYSxx member) in UBBPARM:
     ```plaintext
     SKIP=(PROG=AMUUMAIN,CHECK=(EXCP))
     ```
     Using this method eliminates the need to maintain the code in any subsequent RLST processing.

   - Include the NOCHECK keyword in the specific SMRLSTxx member that is associated with the SPACVOLA function. (The variable `numberOfVolumes` represents the maximum number of volumes that can be available for volume extension.)
     ```plaintext
     SET SPACVOLA=numberOfVolumes NOCHECK=EXCP
     INC PGM=(AMUUMAIN)
     ```
     Using this method instructs the system to allow jobs that execute the listed programs to run regardless of whether those programs use EXCP processing.

Number of DB2 threads that LOADPLUS uses

For most load jobs, the maximum number of batch threads that LOADPLUS uses concurrently is six per job. LOADPLUS uses up to six threads during the UTILINIT phase and two threads for the duration of the job.

The following exceptions apply to specific types of load jobs:

- For an SQLAPPLY load, processing passes to High-speed Apply during the COMBINED phase. High-speed Apply uses a minimum of six threads concurrently, but can use up to two plus the value that you specify for the LOAD command option APMAXAGENTS (or the LOADPLUS installation option APMXAGNT).

- For a DSNUTILB load, DSNUTILB uses additional threads. LOADPLUS does not use any additional threads for a DSNUTILB load.
Load jobs that invoke DSNUTILB

LOADPLUS enables certain features by invoking DSNUTILB. The following topics describe considerations that apply when LOADPLUS invokes DSNUTILB.

**Note**

LOADPLUS invokes DSNUTILB to enable new features quickly. LOADPLUS generally will provide native support for these features (without invoking DSNUTILB) in a later version of the product or via PTF. You cannot use the DSNUTILB option (or any other option) to request that LOADPLUS invoke DSNUTILB for a feature that LOADPLUS supports natively.

LOADPLUS features that require DSNUTILB

For this version, LOADPLUS invokes DSNUTILB to enable loading data when any of the following conditions exists:

- Participating indexes contain a key derived from certain expressions. LOADPLUS natively supports some indexes on expression. For more information, see “Indexes on expression” on page 142.

- Participating indexes contain keys with random ordering.

- Participating indexes are defined with EXCLUDE NULL KEYS.

- You are loading a table that is defined as ORGANIZE BY HASH.

- You are loading a temporal table and the following conditions exist:
  - You specify INDEX UPDATE.
  - A participating index is defined with BUSINESS_TIME WITHOUT OVERLAPS.
  - Any SHRLEVEL value is in effect except SHRLEVEL CHANGE SQLAPPLY.

- You are loading a table that contains columns that are defined with a CCSID specification

- You are loading LOB or XML data in either of the following cases:
  - You are running an SQLAPPLY load.
  - You are running a two-phase load (PRELOAD CONTINUE).
You are loading LOB data in any of the following cases:

— You are loading a partition-by-growth table space.

— You are loading from a referenced file of a different type than the column type (for example, from a BLOB referenced file to a CLOB column).

You are loading XML data in any of the following cases:

— You are loading XML data from a DBCLOB referenced file.

— You are loading an XML column that is defined with an XML schema.

— You are loading a table that contains a user-defined XML index, and the index is defined with DEFINE NO and is not yet materialized.

You are performing certain data translations.
For details about these data translations, see “Data translation” on page 383.

Enabling LOADPLUS to invoke DSNUTILB

You must perform certain steps to enable LOADPLUS to invoke DSNUTILB when needed.

To enable LOADPLUS to invoke DSNUTILB

1 Ensure that you are licensed to use the IBM DB2 LOAD utility.

2 Ensure that the following installation and command options are set:

<table>
<thead>
<tr>
<th>Option</th>
<th>Required setting</th>
<th>More information</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSNUTILB</td>
<td>YES</td>
<td>“DSNUTILB” on page 283 or “DSNUTILB=YES” on page 725</td>
</tr>
<tr>
<td>ACTIVE</td>
<td>YES for the following DDTYPES:</td>
<td>“ACTIVE” on page 398 or “ACTIVE=YES” on page 763</td>
</tr>
<tr>
<td></td>
<td>■ All work file DDTYPES that the load job requires</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ (if you specify COPY YES on a LOAD REPLACE job) The LOCPFCPY DDTYPE and other copy DDTYPES if needed</td>
<td></td>
</tr>
</tbody>
</table>
3 Use the information in the rest of this section to ensure that you create appropriate JCL for your job. For existing jobs, you might need to make changes to your JCL.

4 For additional options or restrictions that might affect your job, see individual procedures for features that use DSNUTILB:
   - “Loading XML data by invoking DSNUTILB” on page 118
   - “Loading LOB data by invoking DSNUTILB” on page 122
   - “Unicode data” on page 123

### General restrictions for a DSNUTILB load

The following general restrictions apply when running a DSNUTILB load:

- LOADPLUS invokes DSNUTILB after the UTILINIT phase. Therefore, most of the processing that LOADPLUS normally does to validate, load, and copy your data is handled by the IBM DB2 LOAD utility for these jobs. Functionality is limited to those features that are available in the DB2 LOAD utility.

- If any table in the table space that you are loading (regardless of whether you are loading that table) uses a feature that LOADPLUS supports via DSNUTILB, LOADPLUS invokes DSNUTILB for that load job.

- Do not specify FREE=CLOSE for your SYSPRINT data set.

- LOADPLUS ignores any SYSPRIN2 DD statement in your JCL.

### Input file for a DSNUTILB load

The following considerations apply to the input file that you use for a DSNUTILB load:

- You can use only a single input file (SYSREC) to load your data.
You can either use the INDSN option to have DSNUTILB dynamically allocate your input file or you can specify the DD statement in your JCL.

**Data set allocation for a DSNUTILB load**

When LOADPLUS invokes DSNUTILB, you must enable dynamic allocation for all work data sets that the load job requires by specifying ACTIVE YES for those DDTYPEs.

If you specify COPY YES, you must also specify ACTIVE YES for your copy data sets. If you specify any copy or work file data sets in your JCL, LOADPLUS ignores them, regardless of your IFALLOC specification.

With the exceptions described in the following table, LOADPLUS uses your dynamic allocation options to generate a TEMPLATE control statement for each data set. LOADPLUS then passes these TEMPLATE statements to DSNUTILB for data set allocation.

<table>
<thead>
<tr>
<th>Command option</th>
<th>Considerations</th>
</tr>
</thead>
</table>
| ACTIVE         | Requires a value of YES for all work file DDTYPEs (and for your copy data sets if you specify COPY YES)  
If ACTIVE is NO, LOADPLUS issues message BMC50179E or BMC50181E and terminates. |
| AVGVOLSP       | Ignores this option |
| DSNPAT         | Converts symbolic variables when an equivalent exists  
If you specify a symbolic variable that does not have an equivalent, you will receive a DSNUTILB error.  
For information about which symbolic variables are converted, see “DSNPAT” on page 414.  
Also, be aware that user-defined variables are not valid for a DSNUTILB load. |
| DSNTYPE        | Ignores this option |
| IFALLOC        | Ignores this option  
If you specify DD statements in your JCL for any of the data sets required for this job, LOADPLUS ignores them. If your JCL contains a data set with the same name as the one that LOADPLUS generates for dynamic allocation, you might encounter a contention error. |
### INTO TABLE statement for a DSNUTILB load

With the exception of the options described in this topic, LOADPLUS passes your INTO TABLE statement as is to DSNUTILB.

Any other options that are invalid for the IBM DB2 LOAD utility will cause a DSNUTILB error and might cause your load job to fail. With the exception of TRIM, your INTO statement’s syntax must be valid for both DB2 LOAD and LOADPLUS.

<table>
<thead>
<tr>
<th>INTO TABLE option</th>
<th>LOADPLUS response if you include this option</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName REPLACE</td>
<td>Terminates</td>
</tr>
<tr>
<td>POS (instead of POSITION) on the field specification</td>
<td>Terminates</td>
</tr>
<tr>
<td>TRIM on the field specification</td>
<td>Translates to the DB2 STRIP specification as follows:</td>
</tr>
<tr>
<td></td>
<td>■ For CHAR or VARCHAR fields, passes as STRIP TRAILING ' '</td>
</tr>
<tr>
<td></td>
<td>■ For BINARY or VARBINARY fields, passes as STRIP</td>
</tr>
</tbody>
</table>
Considerations for other LOADPLUS options when running a DSNUTILB load

For a DSNUTILB load, LOADPLUS responds in one of the following ways when a command or installation option is not valid for the IBM DB2 LOAD utility:

- Ignores the option
- Terminates with a message about the option
- Translates the option to a valid DB2 LOAD option and passes the translated option

In most other cases, LOADPLUS passes options that are valid for both LOADPLUS and the DB2 LOAD utility.

The following table lists the options that are not valid or that are translated, and describes how LOADPLUS responds if your job contains these options. See also the following sections:

- For dynamic allocation options, see “Data set allocation for a DSNUTILB load” on page 60.
- For INTO TABLE, see “INTO TABLE statement for a DSNUTILB load” on page 61.

Table 9: Options that are incompatible or translated for a DSNUTILB load

<table>
<thead>
<tr>
<th>Command or installation option</th>
<th>LOADPLUS response if you include the option</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANALYZE</td>
<td>Issues message BMC50178E and terminates</td>
</tr>
<tr>
<td>APCOLLECTION</td>
<td>Ignores these options</td>
</tr>
<tr>
<td>AP COMMIT</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>APMAXAGENTS</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>APMULTIROW</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>APOWNER</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>APRETRYLIM</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>APRETRYVAL</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>AUTOENUMROWS</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>AVAILPAGEPCT</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>BMC STATS</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>CENTURY</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>CHECKPEND or CHEKPEND</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>COPY YES</td>
<td>Passes COPYDDN and RECOVERYDDN values</td>
</tr>
<tr>
<td>COPY YES REGISTER</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>Command or installation option</td>
<td>LOADPLUS response if you include the option</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>COPYLVL PART</td>
<td>If COPY YES, issues message BMC50178E and terminates</td>
</tr>
<tr>
<td>COPYPEND NO</td>
<td>Passes these options as NOCOPYPEND</td>
</tr>
<tr>
<td>COPYPEND (NO,)</td>
<td></td>
</tr>
<tr>
<td>COPYPEND (NO,ENFORCE)</td>
<td></td>
</tr>
<tr>
<td>DELETEFILES YES or DELFILES YES</td>
<td>Deletes all work files except SYSERR</td>
</tr>
<tr>
<td>DELETEFILES YES SYSDISC YES or DELFILES (YES/NO,YES)</td>
<td>Ignores request to delete SYSDISC</td>
</tr>
<tr>
<td>DISCARDLIMRC</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>DISCARDRC</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>DISCARDS IGNORE</td>
<td>Issues message BMC50178E and terminates</td>
</tr>
<tr>
<td>DISCARDS REPORT</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>DRAIN_WAIT</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>DRNDELAY</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>DRNRETRY</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>DRNWAIT</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>DSNEXIT</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>ENFORCE CONSTRAINTS INFORI</td>
<td>For an SQLAPPLY load, passes ENFORCE CONSTRAINTS</td>
</tr>
<tr>
<td>ENFORCE CHECK CONSTRAINTS</td>
<td>Otherwise, issues message BMC50178E and terminates</td>
</tr>
<tr>
<td>ENFORCE RI</td>
<td></td>
</tr>
<tr>
<td>ENFORCE NO</td>
<td>For an SQLAPPLY load, passes ENFORCE CONSTRAINTS</td>
</tr>
<tr>
<td></td>
<td>Otherwise, passes this option</td>
</tr>
<tr>
<td>ENUMROWS</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>EXCHANGE</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>FASTSWITCH</td>
<td>Causes DSNUTILB to use the value in DSNZPARM</td>
</tr>
<tr>
<td>FILECHK</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>FORCE</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>FORMAT BMC</td>
<td>Issues message BMC50178E and terminates</td>
</tr>
<tr>
<td>FORMAT BMCUNLOAD</td>
<td></td>
</tr>
<tr>
<td>IDCACHE</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>IDCDDN</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>IDERROR DISCARD</td>
<td>Issues message BMC50178E and terminates</td>
</tr>
<tr>
<td>Command or installation option</td>
<td>LOADPLUS response if you include the option</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>IDERROR_FAIL</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>IFDISCARDLIMIT</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>IFDISCARDS</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>IFZEROROWS</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>INDEX</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>INFO</td>
<td>Issues message BMC50178E and terminates</td>
</tr>
<tr>
<td>INLINE or INLINECP</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>INTO TABLE REPLACE</td>
<td>Issues message BMC50178E and terminates</td>
</tr>
</tbody>
</table>
| KEEPDICITIONARY                | For an SQLAPPLY load, ignores this option  
Otherwise, passes KEEPDICITIONARY YES as KEEPDICITIONARY and ignores KEEPDICITIONARY NO |
| LOBAVGPCRT                    | Ignores this option                           |
| LONGNAMETRUNC                  | Ignores this option                           |
| MAXEXTSZ                      | Passes this option as MAXPRIME                 |
| MAXSORTMEMORY                 | Ignores this option                           |
| MAXSORTS                      | Ignores this option                           |
| MAXTAPE                       | Ignores this option                           |
| MINSORTMEMORY                 | Ignores this option                           |
| NLPCTFREE                     | Ignores this option                           |
| ORDER                         | Ignores this option                           |
| ORIGINALDISP                  | Ignores this option                           |
| PREFORMAT                     | NO, ignores this option  
YES, passes as PREFORMAT                             |
| PRELOAD ANALYZE               | Issues message BMC50178E and terminates          |
| PRELOAD PAUSE                 | Ignores this option                           |
| PRELOAD LOAD                  | Ignores this option                           |
| PRELOAD CONTINUE              | Ignores this option                           |
| REDEFINE                      | NO, passes as REUSE  
YES, ignores this option                             |
<p>| RETRY                         | Ignores this option                           |
| RETRY_DELAY                   | Ignores this option                           |</p>
<table>
<thead>
<tr>
<th>Command or installation option</th>
<th>LOADPLUS response if you include the option</th>
</tr>
</thead>
</table>
| REUSE                          | For an SQLAPPLY load, ignores this option  
                                     Otherwise, passes this option                |
| SHORTMEMORY                    | Ignores this option                          |
| SHRLEVEL CHANGE                | For LOAD REPLACE or LOAD RESUME YES PART n REPLACE, issues message BMC50178E and terminates  
                                     For an SQLAPPLY load, processes normally    |
| SHRLEVEL REFERENCE             | Issues message BMC50178E and terminates      |
| SIZEPCT                        | Passes this option as PCTPRIME                |
| SKIPFIELDS                     | Passes this option as IGNOREFIELDS            |
| SKIPPIX                        | Issues message BMC50178E and terminates       |
| SMAX                           | Ignores this option                          |
| SMCORE                         | Ignores this option                          |
| SWITCHTIME                     | Ignores this option                          |
| SYNC                           | Ignores this option                          |
| TOTALPAGEPCT                   | Ignores this option                          |
| TSSAMPLEPCT                    | Ignores this option                          |
| UNIQUECHECK                    | Ignores this option                          |
| UNIQUEINTO NO                  | Ignores this option                          |
| UNIQUEINTO YES                 | Issues message BMC50178E and terminates       |
| UPDATEDB2STATS YES             | For LOAD RESUME NO SHRLEVEL NONE, passes as STATISTICS TABLE (ALL) INDEX (ALL) UPDATE ALL HISTORY ALL FORCEROLLUP YES,  
                                     For other load jobs, ignores this option     |
| UPDATESAXA or UPDMAXA          | Ignores this option                          |
| XBMID                          | Ignores this option                          |
| XMLAVGSIZE                     | Ignores this option                          |
| ZEROROWRC                      | Ignores this option                          |
| ZIIP                           | Ignores this option                          |
LOADPLUS serialization and concurrency

The following topics discuss concurrency issues and object status requirements, which can vary with the command statement specifications.

## Executing BMC utilities concurrently

All BMC Software utility products use the BMCSYNC and BMCUTIL tables to control concurrent access to DB2 objects.

The 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 808.

**Note**

The setting of the LOCKROW installation option determines whether the utility uses MVS enqueues or SQL LOCK TABLE statements to serialize the BMCSYNC and BMCUTIL tables. For information about the LOCKROW option, see the installation options descriptions for the utility.

## 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 808.

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 798). 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 10: 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>
</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. |
<p>| 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. |</p>
<table>
<thead>
<tr>
<th>Product</th>
<th>Access level</th>
<th>Additional information</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOVER PLUS</td>
<td>X, S, or blank</td>
<td>RECOVER PLUS registers an object with shared access (S) under the following conditions:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ The table space for an index is registered with shared access if the index is being rebuilt and its table space is not recovered in the same job.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ A table space partition is registered with shared access if the keys for that partition are unloaded with a RECOVER UNLOADKEYS operation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVER PLUS registers an object with no access status (blank) if you specify the following commands or options:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ The ACCUM command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ OUTCOPY ONLY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ INDEP OUTSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RECOVER PLUS registers the object with exclusive access (X) in all other cases.</td>
</tr>
<tr>
<td>RECOVERY MANAGER</td>
<td>S</td>
<td>None</td>
</tr>
<tr>
<td>REORG PLUS</td>
<td>X</td>
<td>If you specify PART, REORG PLUS registers only the specified partitions with exclusive access (X). If no nonpartitioned indexes exist on the table space, you can run other utilities on different partitions while running this job.</td>
</tr>
<tr>
<td>UNLOAD PLUS</td>
<td>S</td>
<td>None</td>
</tr>
</tbody>
</table>

**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.

---

**Object status when running LOADPLUS**

This topics in this section describes the initial status requirements for each type of load.
The section also describes how LOADPLUS changes the status of the objects during load processing and after load processing completes.

Note

When LOADPLUS invokes DSNUTILB, status checking and changing is handled by DSNUTILB. For information about status handling for these jobs, see the documentation for the LOAD utility in the IBM DB2 Utility Guide and Reference.

Initial object status

The database that is associated with the objects that are participating in the load must have an initial status of read/write (RW).

The allowable statuses for table spaces and index spaces depend on the type of load job that you are running. Use the following table to locate the information for the job that you are running:

Table 11: Status requirements by load type

<table>
<thead>
<tr>
<th>Type of load</th>
<th>Page reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD REPLACE</td>
<td>“LOAD REPLACE or LOAD RESUME YES PART n REPLACE” on page 69</td>
</tr>
<tr>
<td>LOAD RESUME YES PART n REPLACE</td>
<td>“LOAD REPLACE or LOAD RESUME YES PART n REPLACE” on page 69</td>
</tr>
<tr>
<td>LOAD RESUME YES INTO TABLE tableName REPLACE</td>
<td>“LOAD RESUME YES INTO TABLE tableName REPLACE” on page 71</td>
</tr>
<tr>
<td>LOAD RESUME YES (SQLAPPLY load)</td>
<td>“LOAD RESUME YES (SQLAPPLY load)” on page 71</td>
</tr>
<tr>
<td>LOAD RESUME YES (all other)</td>
<td>“LOAD RESUME YES (all other)” on page 71</td>
</tr>
</tbody>
</table>

LOAD REPLACE or LOAD RESUME YES PART n REPLACE

When you specify LOAD REPLACE or LOAD RESUME YES PART n REPLACE, table spaces and index spaces can be in any status except those that the following tables describe:

Table 12: LOAD REPLACE or LOAD RESUME YES PART n REPLACE status restrictions for particular objects

<table>
<thead>
<tr>
<th>Object</th>
<th>Restriction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects that are associated with a table whose table space or index spaces are defined with DEFINE NO and are not already materialized</td>
<td>Must be in RW status and cannot be in any restrictive status</td>
</tr>
</tbody>
</table>
Object indexes on the parent table of the table that you are loading when you are checking referential constraints during the load job

<table>
<thead>
<tr>
<th>Object</th>
<th>Restriction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cannot be in any restrictive status</td>
<td></td>
</tr>
</tbody>
</table>

### Table 13: LOAD REPLACE or LOAD RESUME YES PART n REPLACE status restrictions

<table>
<thead>
<tr>
<th>Status</th>
<th>Restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUXW</td>
<td>Table spaces cannot be in auxiliary warning (AUXW) status.</td>
</tr>
<tr>
<td>DBETE</td>
<td>Objects cannot be in DBET error (DBETE) status.</td>
</tr>
<tr>
<td>LPL or WEPR</td>
<td>If the objects are in LPL or WEPR status, they must be in RW,LPL or RW,WEPR status. For a table space that is in LPL or WEPR status, the LOAD command must include one of the following options:</td>
</tr>
<tr>
<td></td>
<td>- COPY YES with REGISTER for one or more copies</td>
</tr>
<tr>
<td></td>
<td>- COPY NO COPYPEND NO</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If the value of the COPYPEND installation option is (YES,ENFORCE), you cannot specify COPYPEND NO on your LOAD command.</td>
</tr>
<tr>
<td></td>
<td>If a table space that is in LPL or WEPR status contains a parent table of the table that you are loading, you must have the following values for the specified options:</td>
</tr>
<tr>
<td></td>
<td>- The value of CHECKPEND must be NO.</td>
</tr>
<tr>
<td></td>
<td>- The value of ENFORCE must be NO or CHECK CONSTRAINTS.</td>
</tr>
<tr>
<td></td>
<td>An index that is in LPL or WEPR status cannot be in informational COPY pending (ICOPY) status.</td>
</tr>
<tr>
<td></td>
<td>If an index that is in LPL or WEPR status was defined with COPY YES, you must specify COPY NO COPYPEND NO on your LOAD command.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If the value of the COPYPEND installation option is (YES,ENFORCE), you cannot specify COPYPEND NO on your LOAD command.</td>
</tr>
<tr>
<td></td>
<td>You can specify LOAD RESUME YES PART n REPLACE only if no associated nonpartitioned indexes are in LPL or WEPR status.</td>
</tr>
<tr>
<td>PRO</td>
<td>Objects cannot be in persistent read only (PRO) status.</td>
</tr>
<tr>
<td>RESTP</td>
<td>Objects cannot be in restart pending (RESTP) status.</td>
</tr>
<tr>
<td>REORP</td>
<td>If any of the partitions that you specify in a LOAD RESUME YES PART REPLACE job are in REORP status, you must specify all partitions that are in REORP status.</td>
</tr>
<tr>
<td>RO</td>
<td>If you specify SHRLEVEL CHANGE, objects cannot be in read-only (RO) status.</td>
</tr>
</tbody>
</table>
Objects cannot be in read-or-replication-only (RREPL) status.

Objects cannot be in any of the utility restrictive states.

<table>
<thead>
<tr>
<th>Status</th>
<th>Restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>RREPL</td>
<td>Objects cannot be in read-or-replication-only (RREPL) status.</td>
</tr>
<tr>
<td>UTRO</td>
<td></td>
</tr>
<tr>
<td>UTRW</td>
<td></td>
</tr>
<tr>
<td>UTUT</td>
<td></td>
</tr>
</tbody>
</table>

**LOAD RESUME YES INTO TABLE  `tableName` REPLACE**

When you specify LOAD RESUME YES INTO TABLE `tableName` REPLACE, the following status requirements apply:

- The table spaces must adhere to the status requirements for LOAD RESUME YES (described in “LOAD RESUME YES (all other)” on page 71).

- The participating index spaces must adhere to the status requirements for LOAD REPLACE (described in “LOAD REPLACE or LOAD RESUME YES PART n REPLACE” on page 69).

**LOAD RESUME YES (SQLAPPLY load)**

When you specify either of the following options, LOADPLUS performs an SQLAPPLY load:

- LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY
- LOAD RESUME YES SHRLEVEL CHANGE without PART n REPLACE

For this type of load job, LOADPLUS requires that the table spaces and index spaces that are participating in the load have an initial status of RW. In addition, the table spaces and index spaces cannot be in any restrictive status. If any objects participating in the load do not meet these requirements, LOADPLUS issues message BMC50263E and terminates.

**LOAD RESUME YES (all other)**

For all other LOAD RESUME YES load jobs, the table spaces and index spaces can be in any status except the following restrictive or advisory statuses. Table 14 on page 72 describes exceptions for certain objects.

- Auxiliary CHECK pending (ACHKP)
- Advisory restart pending (AREST)
- Auxiliary warning (AUXW)
- DBET error (DBETE)
- Group RECOVER pending (GRECP)
- Logical page list (LPL)
Persistent read only (PRO)
Page set REBUILD pending (PSRBD)
REBUILD pending (RBDP)
Logical part REBUILD pending (RBDP*)
RECOVER pending (RECP)
Refresh pending (REFP)
REORG pending (REORP)
Restart pending (RESTP)
Read-or-replication-only (RREPL)
Utility restrictive state, read-only access allowed (UTRO)
Utility restrictive state, read/write access allowed (UTRW)
Utility restrictive state, utility exclusive control (UTUT)
Write error page range (WEPR)

Table 14: LOAD RESUME YES status restrictions for particular objects

<table>
<thead>
<tr>
<th>Object</th>
<th>Restriction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects that are associated with a table whose table space or index spaces are defined with DEFINE NO and are not already materialized</td>
<td>Must be in RW status and cannot be in any restrictive status</td>
</tr>
<tr>
<td>Primary indexes on the parent table of the table that you are loading when you are checking referential constraints during the load job</td>
<td>Cannot be in any restrictive status</td>
</tr>
</tbody>
</table>

Object status changes

This topic describes how the object status changes during load processing for each type of load.

The statuses that the following table describes apply to all objects that are participating in the load.

Table 15: Summary of status changes during load processing

<table>
<thead>
<tr>
<th>Load type</th>
<th>Initial status</th>
<th>PRELOAD phase</th>
<th>LOAD phase</th>
<th>COMBINED phase</th>
<th>UTILTERM phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD</td>
<td>RW, RO, or UT a b</td>
<td>Unchanged</td>
<td>STOP c</td>
<td>Not applicable</td>
<td>Reset to initial status</td>
</tr>
<tr>
<td>REPLACE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHRLEVEL NONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Two phase</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Load type</td>
<td>Initial status</td>
<td>PRELOAD phase</td>
<td>LOAD phase</td>
<td>COMBINED phase</td>
<td>UTILTERM phase</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------</td>
<td>---------------</td>
<td>------------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>LOAD RESUME YES SHRLEVEL NONE Two phase</td>
<td>RW, RO, or UT</td>
<td>STOP</td>
<td>STOP</td>
<td>Not applicable</td>
<td>Reset to initial status</td>
</tr>
<tr>
<td>SHRLEVEL NONE Single phase</td>
<td>RW, RO, or UT</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>STOP</td>
<td>Reset to initial status</td>
</tr>
<tr>
<td>SHRLEVEL REFERENCE Two phase</td>
<td>RW, RO, or UT</td>
<td>Unchanged</td>
<td>RO</td>
<td>Not applicable</td>
<td>Stop as rename process starts</td>
</tr>
<tr>
<td>SHRLEVEL REFERENCE Single phase</td>
<td>RW, RO, or UT</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>RO</td>
<td>Stop as rename process starts</td>
</tr>
<tr>
<td>LOAD REPLACE SHRLEVEL CHANGE or LOAD RESUME YES SHRLEVEL CHANGE PART n REPLACE Two phase</td>
<td>RW or UT</td>
<td>Unchanged</td>
<td>Unchanged</td>
<td>Not applicable</td>
<td>Reset to initial status after rename or FASTSWITCH process completes</td>
</tr>
</tbody>
</table>
### LOADPLUS serialization and concurrency

<table>
<thead>
<tr>
<th>Load type</th>
<th>Initial status</th>
<th>PRELOAD phase</th>
<th>LOAD phase</th>
<th>COMBINED phase</th>
<th>UTILTERM phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD REPLACE SHRLEVEL CHANGE or LOAD RESUME YES SHRLEVEL CHANGE PART n REPLACE Single phase</td>
<td>RW or UT&lt;sup&gt;a,b&lt;/sup&gt;</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>Unchanged</td>
<td>Reset to initial status after rename or FASTSWITCH process completes</td>
</tr>
<tr>
<td>Any SQLAPPLY load (Always uses single-phase)</td>
<td>RW</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>Unchanged</td>
<td>Unchanged</td>
</tr>
</tbody>
</table>

<sup>a</sup> If the objects are in LPL or WEPR pending status, they must be in RW status.

<sup>b</sup> If you specify SKIPIX SIX or SKIPIX NUSIX, the nonparticipating indexes can be in any initial status. They are placed in a REBUILD pending status (RBDP, RBDP*, or PSRBD) after the load job completes.

<sup>c</sup> STOP status ensures that LOADPLUS has exclusive use of the space while it is being loaded.

<sup>d</sup> When operating on partitioned objects with RESUME YES and the PART REPLACE option specified, LOADPLUS stops and starts only those partitions that are specified in the LOAD command.

### Object status after loading

After the loading of the data is complete, LOADPLUS starts all table spaces and participating index spaces with the same status that they had at the start of the load job.

This status change occurs at different times, depending on the SHRLEVEL option:

- For SHRLEVEL NONE, LOADPLUS changes the status at the beginning of the UTILTERM phase.
- For SHRLEVEL CHANGE (except an SQLAPPLY load) and SHRLEVEL REFERENCE, LOADPLUS changes the status during the UTILTERM phase after registering the LOAD or COPY row in the SYSIBM.SYSCOPY catalog table.
Note

Note the following exceptions:

- If the table space is in both COPY and CHKP statuses, the table space status remains unchanged and LOADPLUS issues message BMC50380I.
- For an SQLAPPLY load, LOADPLUS does not set pending statuses.
- When LOADPLUS invokes DSNUTILB, any status changes are handled by DSNUTILB.

In addition, LOADPLUS takes the actions shown in the following table, depending on the load options:

Table 16: statuses set or reset by LOADPLUS

<table>
<thead>
<tr>
<th>LOAD command option specified</th>
<th>LOADPLUS actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any load</td>
<td>Sets ICOPY for all participating indexes that are defined with COPY YES</td>
</tr>
</tbody>
</table>
| SKIPIX SIX or SKIPIX NUSIX    | Sets all nonpartitioned (SKIPIX SIX) or all non-unique, nonpartitioned (SKIPIX NUSIX) indexes to one of the following REBUILD pending statuses:  
  ■ RBDP  
  ■ RBDP*  
  ■ PSRBD  
  **Exception:** If you specify RESUME YES (without PART REPLACE) and zero rows are loaded during the load job, LOADPLUS leaves the nonparticipating indexes in the same status that they were in at the start of the job. |
| COPY YES                      | Resets the COPY status |
| COPY NO COPYPEND YES<sup>a</sup> | Places the table space in COPY status (with the exceptions described in “COPY” on page 294) |
| COPY NO COPYPEND NO<sup>b</sup> or LOG NO NOCOPYPEND | Does not place the table space in COPY status  
If the table space was in COPY status before the load, LOADPLUS does not reset the status. |
| CHECKPEND YES<sup>c</sup> and either ENFORCE NO or ENFORCE RI | Places the table space in CHKP status if check constraints exist  
For more conditions under which LOADPLUS sets CHKP, including how child table spaces are affected, see “Referential and check constraints” on page 146. |
<table>
<thead>
<tr>
<th>LOAD command option specified</th>
<th>LOADPLUS actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHECKPEND YES (^{c}) and either ENFORCE NO or ENFORCE CHECK CONSTRAINTS</td>
<td>Places the table space in CHKP status if referential constraints exist. For more conditions under which LOADPLUS sets CHKP, including how child table spaces are affected, see “Referential and check constraints” on page 146.</td>
</tr>
<tr>
<td>LOAD REPLACE and the appropriate ENFORCE option based on the type of constraints on the table</td>
<td>Resets CHKP status if all constraints on the table are checked and the table space was in CHKP status before the load. For more conditions under which LOADPLUS resets CHKP, including how partitions and child table spaces are affected, see “Referential and check constraints” on page 146.</td>
</tr>
</tbody>
</table>
| LOAD REPLACE or LOAD RESUME YES PART \(n\) REPLACE | Resets the following statuses of the table space (or partitions being replaced) and participating index spaces if the objects were in that status before the load:  
- AREO*  
- GRECP  
- PSRBD  
- RBDP  
- RBDP*  
- RECP  
- RECP*  
- REFP  
- REORP  
LOADPLUS resets REORP status only if you specify an INTO statement for each table in the table space. |
| LOAD RESUME YES INTO TABLE \(tableName\) REPLACE | Resets the following statuses of the participating index spaces if they were in that status before the load:  
- GRECP  
- PSRBD  
- RBDP  
- RBDP*  
- RECP  
- RECP*  
- REFP  
- REORP |
LOAD command option specified | LOADPLUS actions
---|---
a If the value of the COPYPEND installation option is (NO, ENFORCE), you cannot specify COPYPEND YES on the LOAD command.
b If the value of the COPYPEND installation option is (YES, ENFORCE), you cannot specify COPYPEND NO or NOCOPYPEND on the LOAD command.
c If the value of the CHEKPEND installation option is (NO, ENFORCE), you cannot specify CHECKPEND YES on the LOAD command.

**Drain processing in LOADPLUS**

LOADPLUS can drain tables, table spaces, or partitions, depending on the type of load job that you are running.

The following table shows the types of load jobs for which LOADPLUS drains an object, and the phase in which the drain occurs.

<table>
<thead>
<tr>
<th>Type of load job</th>
<th>Phase in which the drain occurs</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD REPLACE SHRLEVEL NONE two phase</td>
<td>LOAD</td>
</tr>
<tr>
<td>LOAD RESUME YES SHRLEVEL NONE two phase</td>
<td>PRELOAD</td>
</tr>
<tr>
<td>SHRLEVEL NONE single phase</td>
<td>COMBINED</td>
</tr>
<tr>
<td>SHRLEVEL REFERENCE</td>
<td>PRELOAD or COMBINED (before starting the object RO)</td>
</tr>
<tr>
<td>LOAD REPLACE SHRLEVEL CHANGE or LOAD RESUME YES SHRLEVEL CHANGE PART n REPLACE</td>
<td>UTILTERM (when the rename process starts)</td>
</tr>
</tbody>
</table>

Typically, the drain is held for a very short period of time, while LOADPLUS stops the object. However, you can use the following command options to control drain behavior:

- DRAIN_WAIT
- RETRY
- RETRY_DELAY
- FORCE

If LOADPLUS cannot drain all of the objects within the time period specified by DRAIN_WAIT, LOADPLUS releases the drains that it has obtained so far. LOADPLUS waits for the length of time that you specified in the RETRY_DELAY command option and tries to drain the objects again. LOADPLUS retries for the number of times that you specified in the RETRY command option. The FORCE
option can change this process by enabling you to specify if and when you want to cancel any threads that might prevent a successful drain.

Note
You can also use installation options to set defaults for these command options.

Considerations when loading multi-data-set DB2 objects

The following requirements and considerations apply to multi-data-set objects:

- When loading multi-data-set table spaces and index spaces that are VCAT-defined, you must complete one of the following tasks. If you do not complete one of these tasks, the LOADPLUS job fails.
  - Predefine the required data sets.
  - Specify REDEFINE YES and include a SYSIDCIN data set that defines the required data sets.

- For STOGROUP-defined multi-data-set objects, LOADPLUS defines additional data sets if needed for expansion, even if the value of the REDEFINE option is NO.

- The PRELOAD phase of a two-phase load uses the data set allocations, the number of extents allowed, and DB2 restrictions to determine whether the DB2 object can hold the data that you are loading. If the DB2 object cannot hold the data, the PRELOAD phase issues message BMC51487E and terminates. However, the COMBINED phase of a single-phase load cannot make this prediction due to real-time loading of data. As a result, if the DB2 object is unable to hold the data that you are loading, LOADPLUS fails with an extent failure or similar condition.

Dynamic data set allocation in LOADPLUS

LOADPLUS enables you to dynamically allocate the data sets that are needed during the load job.

You activate dynamic data set allocation through command syntax or installation defaults. When dynamic allocation is active, LOADPLUS calculates the optimal size and number of data sets and allocates them for you. Dynamic allocation reduces or
eliminates the need to specify DD statements for these files in your JCL. LOADPLUS also optionally deletes the work files.

With dynamic allocation, you spend less time performing analysis to set up optimized JCL for LOADPLUS jobs. Also, you will not need to modify the JCL for the LOADPLUS job as DB2 objects change size or structure over time.

In a two-phase load, dynamic allocation takes place at the beginning of the PRELOAD phase. For a single-phase load, this allocation occurs at the beginning of the COMBINED phase.

The topics in this section describe the standard dynamic data set allocation process and options for most situations. The following special conditions apply:

- You can also have LOADPLUS dynamically allocate your input (SYSREC) data sets by using the INDSN command option. However, LOADPLUS does not analyze requirements for these data sets and does not use the standard LOADPLUS dynamic allocation options that this section describes. The considerations in this section do not apply to input data sets. For more information, see “INDSN” on page 211.

- When LOADPLUS invokes DSNUTILB, you must enable dynamic allocation for the required data sets. You can use some of the options described in this section to control dynamic allocation of these data sets, but DSNUTILB handles the allocation.

For information about allocating a particular data set type, see “LOADPLUS DD statements” on page 432.

### Enabling dynamic allocation in LOADPLUS

To enable and use dynamic data set allocation quickly and simply, specify the following options:

- Enable automated file size estimation either in your installation options (AUTOENUMROWS=YES) or on your LOAD command (ENUMROWS AUTO).

- For any of the DDTYPES that you want to dynamically allocate, specify ACTIVE YES on your LOAD command or in your installation options.

For more options that you can use with dynamic allocation, see “ENUMROWS” on page 229 and “Dynamic allocation options” on page 393.
Using dynamic allocation in a worklist environment

When running in a worklist environment, the utility ignores the ACTIVE option in your installation options module. The utility dynamically allocates your data sets only if the invoking product (DASD MANAGER PLUS, CATALOG MANAGER, or CHANGE MANAGER) supplies the ACTIVE YES syntax.

Generating data set names in LOADPLUS

You can use the DSNPAT option to specify a pattern to generate a unique data set name.

Names created with DSNPAT

The data set name pattern (DSNPAT) option allows you to specify text and variable data for building data set names. If you cannot construct a data set name that meets your organization’s standards by using text and the supplied variables, LOADPLUS provides an exit point that allows you to create your own variables for use with DSNPAT. Sample exits written in assembler, COBOL, C, and LE C are described in “LOADPLUS user exits” on page 825 and are provided in the HLQ.LLQSAMP library. For more information about the DSNPAT option, see “DSNPAT” on page 414.

The pattern that you specify in your DSNPAT option must allow LOADPLUS to generate unique data set names. For multiple SORTOUT and SYSUT1 files, you must include the &DDNAME variable to generate unique names. For copy data sets, you might need to include additional variables, such as &VCAT, &DATEJ, or &TIME4, to generate unique names across multiple loads. If LOADPLUS encounters non-unique data set names, the job terminates.

GDG names

You can use generation data group (GDG) names for the following types of dynamically allocated files:

- Copy data sets
- SYSDISC files
- SYSERR files
- SYSMAP files

Each DDTYPE must have a different GDG base.

GDG name format
The GDG format that you use to construct data set names is the same as the format that you use in JCL to allocate data sets through DD statements: you append the generation number in parentheses. The open parenthesis tells LOADPLUS that the pattern is a GDG name. The generation number must be an integer from 1 through 255.

An example of a GDG name is &UTILPFX.&DDNAME.(+1). If you are using a substitution variable as the last variable before the open parenthesis, you must include a period before the open parenthesis.

**GDG base**

LOADPLUS has the following requirements for the number of GDG bases that you specify:

- Each DDTYPE must have a different GDG base.
- For copy data sets, each partition must have a different GDG base if you specify COPYLVL PART on the LOAD command.

If the base does not exist, LOADPLUS creates it for you, using everything in the pattern up to the open parenthesis as the base name.

When defining the base, LOADPLUS uses the values of the following options:

- The GDGLIMIT installation or command option allows you to specify the number of generations to keep.
- If the GDGLIMIT value is exceeded, the GDGEMPTY installation option tells the system to uncatalog either all preexisting generations of this data set or only the oldest generation.
- The GDGSCRATCH installation option tells the system whether to delete the entry that was just uncataloged from the volume’s table of contents (VTOC).

For more information, see the descriptions for the dynamic allocation installation options, and the command option description for “GDGLIMIT” on page 422.

**Specifying ddname prefixes**

If you specify more than one ddname prefix for dynamic allocation, the prefix for each ddname must be different enough for the product to differentiate one prefix from another.

To be different enough, if these prefixes are different only because one prefix has additional trailing bytes, then these trailing bytes must contain at least one
nonnumeric byte. For example, the first set of prefixes that follow is sufficiently
different, but the second set is not:

- Acceptable set:
  - BMCRD
  - BMCRDWK

- Not acceptable set:
  - BMCRD
  - BMCRD11

The prefixes that you specify must allow the utility to add the data set number (or
partition number in the case of copy data sets) and still result in a valid ddname of
eight characters. If the generated name would result in a ddname of less than eight
characters, the utility pads the data set or partition number with leading zeros. The
following table shows an example of how the prefixes that you specify resolve to the
generated ddnames.

Table 18: Example of specifying ddname prefixes

<table>
<thead>
<tr>
<th>Prefix specified for SYSUT1 data sets</th>
<th>Number of data sets</th>
<th>Generated ddnames</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSOUT1</td>
<td>9</td>
<td>SYSOUT1...SYSOUT9</td>
</tr>
<tr>
<td>SYSOUT1</td>
<td>10</td>
<td>None (invalid length)</td>
</tr>
<tr>
<td>WORK</td>
<td>10</td>
<td>WORK0001...WORK0010</td>
</tr>
</tbody>
</table>

Deleting dynamically allocated data sets in LOADPLUS

You can choose to delete dynamically allocated data sets automatically after each
successful job, or you can manually delete them.

Note
LOADPLUS also provides options that allow you to specify an expiration date or
retention period for certain dynamically allocated data sets. For more information,
see “EXPDT” on page 420 and “RETPD” on page 421.

Automatically deleting after successful jobs

To delete dynamically allocated data sets, specify DELETEFILES YES on your LOAD
command. After the load job completes successfully, LOADPLUS automatically
deletes the work files that it dynamically allocated and those allocated in your JCL,
except SYSDISC. DELETEFILES YES does not apply to image copy data sets that
LOADPLUS dynamically allocates.

You can also have LOADPLUS delete SYSDISC, if no records were written to this
data set during the load process, by specifying SYSDISC YES. If LOADPLUS wrote
records to SYSDISC during the load process, LOADPLUS does not automatically delete SYSDISC, allowing you to correct the discarded records that it contains and resubmit the job. You must manually delete SYSDISC in this case.

You can also specify these preferences with the DELFILES installation option. For more information about these options, see “DELETEFILES” on page 250 or “DELFILES=(YES,NO)” on page 720.

Deleting manually

The SYSPRINT from your LOADPLUS job contains a report of the dynamically allocated work files. When you need to manually delete work files, you can use this report to determine which files to delete.

Dynamically allocating larger data sets with different properties than smaller data sets

You can use the THRESHLD option and associated dynamic allocation options to tell the utility to use different properties for larger data sets than for smaller ones. Data set allocations that exceed the threshold value will use the values for the second parameter of applicable dynamic allocation options.

For example, you could use the following options to tell the utility to send data sets greater than 720 MB to tape device TAPE1, and smaller data sets to DASD device SYSDA:

```
UNIT(SYSDA,TAPE1)
THRESHLD 720000
```

For more information, see the THRESHLD option description.

Using SMS ACS routines to influence dynamic allocation

If your SMS automatic class selection (ACS) routines use the UNIT parameter to influence data set allocation, you can use the SMSUNIT option in the utility to affect that use.

The value of the SMSUNIT option affects the UNIT value as follows:

- When you specify SMSUNIT YES, the utility passes the UNIT option to SMS allocation in addition to passing the SMS class options and other normally passed options.
When you specify SMSUNIT NO, the utility does not pass the UNIT option.

**Reaching the MAXTAPE limit during dynamic allocation**

When UNIT and THRESHLD specifications require that the utility dynamically allocate tape units, allocation occurs in the following priority order:

1. The utility attempts to allocate the greatest number of tape units required that will optimize multitasking.

2. If this number of tape units exceeds the MAXTAPE value, the utility decreases the multitasking level until the number of tape units required is less than or equal to the MAXTAPE value.

   This action might result in the utility dynamically allocating a single SYSUT1 data set, rather than one data set for each non-data-sorting index (thus decreasing multitasking).

3. If the minimum number of tape units required exceeds the MAXTAPE value, the utility issues a message and terminates.

   The value that you specify for the MAXTAPE option includes the units that are required for full and incremental copy data sets.

**Changing dynamic allocation options on restart in LOADPLUS**

Before restarting a job, you might need to change the options that affect dynamic allocation.

For example, if specifying an invalid UNIT or overly restrictive MAXTAPE value causes the job to terminate, then you need to change the relevant option before restarting the job.

The following restrictions apply to changes that you make to dynamic allocation options before restarting a job:

- You cannot change the value for the ACTIVE option on any restart.

- Changing any option on restart such that it results in different ddnames or a different number of DDs than the original option can produce an error. If you need to change the number of SORTOUT or SYSUT1 work files, resubmit the job with a parameter of NEW.

- If you change the value of other dynamic allocation options, specify RESTART(PHASE).
Work file validity and integrity checks in LOADPLUS

LOADPLUS ensures that the work files it uses for load processing are valid by performing the following types of file verification:

- **Check for data set attributes**
  The first check determines whether a work file has been allocated as a temporary data set or with DISP=MOD.

- **Check for data integrity**
  The second check ensures that the work file that LOADPLUS reads during the LOAD phase is the same one that was created during the PRELOAD phase.

Check for data set attributes

This topic describes how LOADPLUS responds when determining that your work files and copy data sets are temporary data sets.

LOADPLUS defines a temporary data set as one whose normal or abnormal DISP is defined with one of the following values:

- **DELETE**
- **NEW,PASS**
- **OLD,PASS** and the original status is not OLD

**SORTOUT, SYSERR, and SYSUT1 data sets**

If LOADPLUS determines that your SORTOUT, SYSERR, or SYSUT1 data sets are temporary, LOADPLUS responds based on the value of the FILECHK installation option:
### FILECHK value | LOADPLUS response
--- | ---
FAIL | If your installation options specify FILECHK=FAIL and if LOADPLUS finds that any of the SORTOUT, SYSERR, or SYSUT1 data sets are temporary, LOADPLUS terminates. On output, if the disposition is MOD, LOADPLUS resets it to empty.
If LOADPLUS is restarting in the LOAD phase and the data set is temporary due to its normal disposition, LOADPLUS overrides the normal disposition to KEEP to ensure against data loss after termination.

WARN | You can force LOADPLUS to process the SORTOUT, SYSERR, and SYSUT1 files in WARN mode by specifying FILECHK=WARN in the LOADPLUS installation options. In WARN mode, LOADPLUS issues a warning message for each work file that is allocated as a temporary data set, but continues load processing.

**Note:** If you restart your job after receiving this warning message, unpredictable errors can occur due to invalid data in the work data set.

### SYSDISC and copy data sets

If LOADPLUS finds that the SYSDISC work file or any of the copy data sets are temporary data sets, LOADPLUS continues processing and issues a warning message, regardless of the value of FILECHK.

**Note**
LOADPLUS issues a warning message for the copy data sets when opening them during the LOAD phase. If you run LOADPLUS in two steps by using the PRELOAD PAUSE option, you can code the DDs with DUMMY in the PRELOAD phase to avoid receiving message BMC50391E. This message states that LOADPLUS is unable to locate the copy data set.

### Check for data integrity

LOADPLUS performs a second check of the SORTOUT, SYSERR, and SYSUT1 work files to ensure data integrity.

### Two-phase load

During the PRELOAD phase of two-phase load processing, LOADPLUS creates the SORTOUT, SYSERR, and SYSUT1 work files with header information about the current utility execution. During the LOAD phase, LOADPLUS checks this header information to ensure that the work file is the file it is expecting to load. If the header information does not match, LOADPLUS terminates.
Single-phase load

During the COMBINED phase of single-phase load processing, LOADPLUS writes and verifies the header information of the SYSERR file.

For certain single-phase load jobs, if SORTOUT and SYSUT1 files are available, LOADPLUS writes the header information during the COMBINED phase of the original load job. If you restart this job, LOADPLUS verifies the header information during the COMBINED phase of the restarted job. If the header information does not match, LOADPLUS terminates and issues an error message. For information about when LOADPLUS uses SORTOUT and SYSUT1 files in a single-phase load, see “SORTOUT data sets in LOADPLUS” on page 443 and “SYSUT1 data sets in LOADPLUS” on page 459.

SHRLEVEL considerations in LOADPLUS

The SHRLEVEL option specifies the level of access that DB2 has to the objects that you are loading during LOADPLUS processing.

- With SHRLEVEL NONE, the objects that you are loading are stopped and unavailable during load processing.
- With SHRLEVEL REFERENCE, the objects that you are loading are available in read-only status during load processing.
- With SHRLEVEL CHANGE, the objects that you are loading are available in RW status during load processing.

Use the following table to determine which SHRLEVEL option to specify:

Table 19: Determining which SHRLEVEL option to use

<table>
<thead>
<tr>
<th>Type of access to your data during the load process</th>
<th>Additional site or application requirements</th>
<th>SHRLEVEL option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do not need access</td>
<td>Not applicable</td>
<td>SHRLEVEL NONE</td>
</tr>
<tr>
<td>Type of access to your data during the load process</td>
<td>Additional site or application requirements</td>
<td>SHRLEVEL option</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>---------------------------------------------</td>
<td>-----------------</td>
</tr>
</tbody>
</table>
| Read-only access                                  | ■ Want very limited outage (which occurs during the rename or FASTSWITCH process)  
■ Want to replace existing data                    | LOAD REPLACE SHRLEVEL REFERENCE |
|                                                   | ■ Want very limited outage (which occurs during the rename or FASTSWITCH process)  
■ Do not want to replace existing data              | LOAD RESUME YES SHRLEVEL REFERENCE |
| Read/write access                                 | ■ Want very limited outage to the table space (LOAD REPLACE) or partition (LOAD RESUME YES PART \( n \) REPLACE)  
■ Want a full replacement of the data in the table space or partition  
   Full replacement includes replacing inserts, updates, and deletes to the base table space or partition that occur during the load process. | LOAD REPLACE SHRLEVEL CHANGE  
or  
LOAD RESUME YES SHRLEVEL CHANGE PART \( n \) REPLACE |
|                                                   | ■ Want no outage to the table space         | SQLAPPLY load (LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY) |
|                                                   | ■ Want to retain all insert, update, and delete operations that affect the table during the load process  
■ Do not want to replace existing data              | |

**SHRLEVEL considerations in LOADPLUS**

88  
LOADPLUS for DB2 Reference Manual
The rest of the topics in this section discuss considerations for the SHRLEVEL REFERENCE and SHRLEVEL CHANGE options. For more information about SHRLEVEL topics, see the following references:

<table>
<thead>
<tr>
<th>Additional SHRLEVEL information</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB2 and system authority</td>
<td>“System setup for LOADPLUS” on page 49</td>
</tr>
<tr>
<td>Object statuses</td>
<td>“Object status when running LOADPLUS” on page 68</td>
</tr>
<tr>
<td>Command syntax information</td>
<td>“SHRLEVEL” on page 189</td>
</tr>
<tr>
<td>Installation option information</td>
<td>“SHRLEVEL=(NONE,NONE)” on page 747</td>
</tr>
<tr>
<td>Recovering from a failure or termination</td>
<td>“Recovering the DB2 object after terminating or canceling a LOADPLUS job” on page 478</td>
</tr>
</tbody>
</table>

## Running SHRLEVEL REFERENCE

When you specify SHRLEVEL REFERENCE, LOADPLUS places the object that you are loading in read-only status while it loads the data to a staging data set.

After the load completes, LOADPLUS replaces the original object with the staging data set.

During the load process, the original object is available in read-only status. During the brief rename or FASTSWITCH process, which occurs in the UTILTERM phase, the object is stopped. This nondestructive process leaves the original data sets intact throughout the load process and allows you to easily restart from a failure or make the objects available without having to recover the objects.

### Before you begin

Ensure that the job that you are going to run will not invoke DSNUTILB. If you specify SHRLEVEL REFERENCE for a load job that invokes DSNUTILB, LOADPLUS issues message BMC50178E and terminates.

For LOAD RESUME YES, LOADPLUS uses the Instant Snapshot capability of XBM or SUF to create the staging data set. Ensure that you meet the following requirements:

- You must be using one of the intelligent storage devices that XBM or SUF supports (as documented in the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide).

- You must have sufficient DASD available to create an Instant Snapshot of the table space or partitions that you are loading and the participating indexes.
The storage group on which the table space is defined must contain volumes from a single vendor, and all volumes in that storage group must support Instant Snapshots.

To run a SHRLEVEL REFERENCE load job

1 (LOAD RESUME YES only) Ensure that the intelligent storage device that you are using for the Instant Snapshot is set up and running correctly.

For details, see the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide.

2 (LOAD RESUME YES only) Ensure that XBM or SUF is available and enabled:
   a Ensure that a currently supported version of XBM or SUF (as described in “LOADPLUS system and software requirements” on page 49) is installed and available to LOADPLUS.
      XBM and SUF require a started task.
   b Set the appropriate SSI options in XBM or SUF, as described in the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide.
   c Create and activate the XBM management set that contains the appropriate snapshot template.
      For more information, see the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide.
   d (optional) Use the XBMID installation or command option to specify the XBM subsystem that you want to use for snapshot processing.

3 Depending on the value in effect for the REDEFINE option, complete any staging data set tasks that are required as described in “Staging data sets” on page 98.

4 Specify SHRLEVEL REFERENCE.

5 Determine whether to use the FASTSWITCH or rename process, and specify FASTSWITCH YES if applicable.

   For information about these processes, see “Staging data sets” on page 98. For information about the FASTSWITCH option, see “FASTSWITCH” on page 253.

6 If necessary, specify a value for the ORIGDISP command option or ORIGDISP installation option.

   For more information about these options, see “ORIGDISP” on page 252 or “ORIGDISP=DELETE” on page 741.
Specify a value for the SWITCHTIME option.

For information about this option, see “SWITCHTIME” on page 255.

**SHRLEVEL CHANGE considerations**

You can specify SHRLEVEL CHANGE with LOAD REPLACE, LOAD RESUME YES PART n REPLACE, or LOAD RESUME YES (without PART n REPLACE). The load process differs based on these options.

**Running LOAD REPLACE or LOAD RESUME YES PART n REPLACE**

When you specify SHRLEVEL CHANGE with LOAD REPLACE or LOAD RESUME YES PART n REPLACE, the table space or partition that you are loading remains in read/write status while LOADPLUS loads the data to a staging data set.

After the load completes, LOADPLUS replaces the original object with the staging data set.

During the load process, the original object is available in read/write status. The original data sets are intact throughout the load process, allowing you to easily restart from a failure or make the objects available without having to recover the objects.

**WARNING**

This type of load fully replaces the data in the table space (or partition), including any inserts, updates, or deletes to the base object that occur during the load process.

**To run a SHRLEVEL CHANGE job with LOAD REPLACE or LOAD RESUME YES PART n REPLACE**

1. Depending on the value for your REDEFINE command or installation option, complete any staging data set tasks that are required as described in “Staging data sets” on page 98.

2. Specify SHRLEVEL CHANGE.

3. Determine whether to use the FASTSWITCH or rename process, and specify FASTSWITCH YES if applicable.

For information about these processes, see “Considerations when loading Unicode data” on page 124. For information about the FASTSWITCH option, see “FASTSWITCH” on page 253.
4 If necessary, specify a value for the ORIGINALDISP command option or ORIGDISP installation option.

For more information about these options, see “ORIGINALDISP” on page 252 or “ORIGDISP=DELETE” on page 741.

5 (optional) Specify a value for the SWITCHTIME option.

For information about this option, see “SWITCHTIME” on page 255.

**Restrictions on SHRLEVEL CHANGE LOAD REPLACE**

The following restrictions apply when you specify SHRLEVEL CHANGE and you are replacing one or more partitions.

LOADPLUS terminates when any of the following conditions exists and you specify SHRLEVEL CHANGE with LOAD REPLACE:

---

**Note**

See also the restrictions on the REPLACE option (“REPLACE” on page 187).

---

- You are running a load job that invokes DSNUTILB. LOADPLUS issues message BMC50178E and terminates with return code 8.

- Partitions are added to a partition-by-growth table space (other than by LOADPLUS) while you are loading that table space. LOADPLUS issues messages BMC50225E and BMC50010U and terminates with return code 16.

LOADPLUS terminates when any of the following conditions exists and you specify SHRLEVEL CHANGE with LOAD RESUME YES PART n REPLACE:

---

**Note**

See also the restrictions on the PART n REPLACE option (“INTO TABLE” on page 306).

---

- Nonpartitioning indexes exist on the table space.

- You are loading a partition-by-growth table space.

- You are running a load job that invokes DSNUTILB. LOADPLUS issues message BMC50178E and terminates with return code 8.
Running LOAD RESUME YES (SQLAPPLY load)

When you specify either of the following options, LOADPLUS performs an SQLAPPLY load:
- LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY
- LOAD RESUME YES SHRLEVEL CHANGE without PART n REPLACE

For an SQLAPPLY load, the object that you are loading remains in read/write status while LOADPLUS loads the data by using SQL insert processing.

For this type of load, LOADPLUS uses the High-speed Apply Engine component of the BMC Log Master for DB2 product. This feature does not require that you have a license for the Log Master product. However, the High-speed Apply Engine must be installed and made available through the STEPLIB, JOBLIB, or LINKLIST at runtime. The High-speed Apply Engine is installed separately from the Log Master product.

To run an SQLAPPLY load job

1. Ensure that a minimum of version 11.1.00 of the High-speed Apply Engine is available to LOADPLUS.
2. Ensure that you have the appropriate authorizations.

   For information about these authorizations, see “Setting LOADPLUS authorizations” on page 51.
3. With your LOAD command, specify one of the following syntax options or the equivalent installation option:
   - RESUME YES SHRLEVEL CHANGE SQLAPPLY
   - RESUME YES SHRLEVEL CHANGE without PART n REPLACE

   Optionally, specify one or more of the options that are available with SHRLEVEL CHANGE SQLAPPLY. For information about these options, see “SHRLEVEL” on page 189.
4. Include an APTDUMP DD statement in your JCL.

   If High-speed Apply generates a SNAP dump in response to an error, High-speed Apply writes the dump to this data set. For more information, see “APTDUMP data sets in LOADPLUS” on page 435.
5. Use the information in “Considerations for an SQLAPPLY load” on page 94 to make any necessary changes to your JCL or environment.
Considerations for an SQLAPPLY load

The following considerations apply when you are running an SQLAPPLY load job (SHRLEVEL CHANGE LOAD RESUME YES).

**LOB and XML data**

LOADPLUS invokes DSNUTILB when you use SQLAPPLY to load LOB or XML data.

**DSNUTILB load jobs**

The following table describes the restrictions that apply to certain options when invoking DSNUTILB for an SQLAPPLY load job:

<table>
<thead>
<tr>
<th>Option</th>
<th>Restriction</th>
</tr>
</thead>
<tbody>
<tr>
<td>APCOLLECTION</td>
<td>Ignores these options</td>
</tr>
<tr>
<td>APCOMMIT</td>
<td></td>
</tr>
<tr>
<td>APMAXAGENTS</td>
<td></td>
</tr>
<tr>
<td>APMULTIROW</td>
<td></td>
</tr>
<tr>
<td>APOWNER</td>
<td></td>
</tr>
<tr>
<td>APRETRYLIM</td>
<td></td>
</tr>
<tr>
<td>APRETRYVAL</td>
<td></td>
</tr>
<tr>
<td>ENFORCE CONSTRAINTS INFORI</td>
<td>Passes these options as ENFORCE CONSTRAINTS</td>
</tr>
<tr>
<td>ENFORCE CHECK CONSTRAINTS</td>
<td></td>
</tr>
<tr>
<td>ENFORCE RI</td>
<td></td>
</tr>
<tr>
<td>ENFORCE NO</td>
<td></td>
</tr>
<tr>
<td>KEEPDICIONARY</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>REUSE</td>
<td>Ignores this option</td>
</tr>
</tbody>
</table>

**DB2 logs**

Because this type of load uses insert processing, it writes to your DB2 logs unless your table space is defined as NOT LOGGED. With this additional activity, you might need to increase the size of your active logs. Refer to your IBM DB2 installation guide for guidelines to use when sizing your active logs.
Processing phases

For this type of load, LOADPLUS uses its single-phase load architecture to pass your data to High-speed Apply.

Work data sets

For this type of load, LOADPLUS does not require the SORTOUT and SYSUT1 data sets, and does not use them if they are specified.

SNAP dumps

When High-speed Apply generates a SNAP dump in response to an error, High-speed Apply tells the dump to an APTDUMP data set.

To enable this capability, you must include the following DD statement in your LOADPLUS JCL:

```
//APTDUMP DD SYSOUT=*  
```

Table space organization

High-speed Apply loads your data by using SQL INSERT statements. Therefore, following an SQLAPPLY load, your table functions as a table whose data was inserted, not loaded. Because insert processing does not preserve the table’s free space and free pages, this type of load can affect the organization of your table. Consider reorganizing your table space following this type of load.

ORDER YES can help maintain the order of your table space and reduce the need to reorganize the table space. Specify ORDER YES if both of the following conditions exist:

- You are adding new rows to an empty table space or to a table space with little or no free space.
- You want the new records inserted in clustering key or partitioning key order.

Multiple table loads

For performance reasons with this type of load, LOADPLUS sorts multiple table data only if you specify ORDER YES.

Referential integrity

If you are loading multiple tables that have foreign key relationships between them, and you want your parent tables to load before their child tables, specify ORDER
YES. This causes LOADPLUS to assign a table number based on the parent-child relationships and to sort your data by this assigned table number.

If you are loading multiple tables that are self-referential or that have circular relationships, LOADPLUS cannot determine the correct order. To maintain referential integrity in this case, order the data before running your load job, and specify ORDER NO on your LOAD command.

*Note*
An SQLAPPLY load uses DB2 to enforce referential integrity. Therefore, LOADPLUS ignores the ENFORCE option for an SQLAPPLY load.

**Duplicate keys**

When you run an SQLAPPLY load, LOADPLUS handles duplicates differently from other load types. For this type of load, if the input file contains duplicate rows, LOADPLUS loads the first one that it encounters and discards any subsequent ones.

*Note*
The record that LOADPLUS loads and the duplicate records that LOADPLUS discards might not be the ones that you expect. For example, if you specify ORDER YES, the sequence of the records that LOADPLUS loads might be different from the sequence of the records in your input file.

This type of load uses DB2 insert processing and DB2 rejects a duplicate row in the first index in which it detects that the row is a duplicate. Therefore, LOADPLUS might not detect all indexes in which a row is a duplicate.

For this type of load, LOADPLUS detects and discards duplicates during insert processing in the COMBINED phase.

**Copies**

For an SQLAPPLY load, your table space is always available. Therefore, for this type of load, LOADPLUS does not create an image copy or DSN1COPY-type copy. If you include COPY YES in your LOAD command, LOADPLUS terminates.

**Restart**

If you restart an SQLAPPLY load, you must use the same input data that you used originally and it must be in the same order. For example, if you are loading concatenated data sets, these data sets must be in the same order as they were originally, and you must not include any additional data sets. Similarly, if LOADPLUS encountered duplicates during the original job, you should not remove them before you restart.
**WARNING**

Data that is missing or that is in a different order in the restarted job than in the original job produces unpredictable results.

**Statistics**

For an SQLAPPLY load, LOADPLUS does not update BMC statistics tables or DB2 catalog statistics. However, because this type of load uses SQL insert processing, DB2 updates the real-time statistics tables as it would for any other DB2 application that uses SQL insert processing.

For an SQLAPPLY load, LOADPLUS generates a report that indicates the number of loaded rows and the number of rows that were discarded due to SQL errors. LOADPLUS provides this information by table space, partition, and table.

**Note**

For partition-by-growth table spaces, this report does not indicate how many rows were loaded into each partition. Use the information in this message only to ensure that the total number of rows loaded into the table space is correct.

**Incompatible LOADPLUS command options**

Some LOADPLUS command options are not available when you perform an SQLAPPLY load. The following table describes the options that are not valid with this type of load and how LOADPLUS responds if your job contains these options:

<table>
<thead>
<tr>
<th>Command option</th>
<th>LOADPLUS response</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMCSTATS</td>
<td>Ignores this option and issues message BMC50109I</td>
</tr>
<tr>
<td>CHECKPEND YES</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>COPY NO COPYPEND YES</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>COPY YES</td>
<td>Issues message BMC50115E and terminates</td>
</tr>
<tr>
<td>ENFORCE</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>IDCACHE</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>IDENTITYOVERRIDE</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>IDERROR</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>INDEX BUILDINDEX UPDATE</td>
<td>Ignores these options</td>
</tr>
<tr>
<td>LOBAVGPC</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>LOG NO</td>
<td>Issues message BMC50115E and terminates</td>
</tr>
<tr>
<td>Command option</td>
<td>LOADPLUS response</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>MAXSORTS</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>NLPCFTFREE</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>ORDER PRESORTED</td>
<td>Issues message BMC50115E and terminates</td>
</tr>
<tr>
<td>ORDER YES ASSOCIATE BYCLUSTERKEY</td>
<td>If referential integrity (RI) exists between the tables that you are loading, changes this option to ASSOCIATE BYTABLE (and issues message BMC50138I) Otherwise, ignores this option</td>
</tr>
<tr>
<td>PART n REPLACE</td>
<td>Issues message BMC50115E and terminates</td>
</tr>
<tr>
<td>PART PREFORMAT</td>
<td>Ignores these options and issues message BMC50109I</td>
</tr>
<tr>
<td>PREFORMAT</td>
<td>Ignores these options and issues message BMC50109I</td>
</tr>
<tr>
<td>PRELOAD LOAD</td>
<td>Ignores these options and issues message BMC50109I</td>
</tr>
<tr>
<td>PRELOAD CONTINUE</td>
<td>LOADPLUS always uses single-phase processing for an SQLAPPLY load.</td>
</tr>
<tr>
<td>PRELOAD PAUSE</td>
<td>Issues message BMC50115E and terminates</td>
</tr>
<tr>
<td>PRELOAD ANALYZE</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>REDEFINE</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>TABLE tableName REPLACE</td>
<td>Issues message BMC50115E and terminates</td>
</tr>
<tr>
<td>TSSAMPLEPCT</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>UNIQUECHECK</td>
<td>Ignores this option</td>
</tr>
<tr>
<td>UPDATEDB2STATS</td>
<td>Ignores this option and issues message BMC50109I</td>
</tr>
<tr>
<td>XMLAVGSIZE</td>
<td>Ignores this option</td>
</tr>
</tbody>
</table>

## Staging data sets

When you specify SHRLEVEL CHANGE (except an SQLAPPLY load) or SHRLEVEL REFERENCE, LOADPLUS uses staging data sets for shadow copies.

LOADPLUS writes the loaded objects to the staging data sets instead of to the original data sets. At the end of the job, LOADPLUS replaces the original data sets with the staging data sets by using either the rename process or the FASTSWITCH process. The following sections describe these processes.

Whether LOADPLUS renames the data sets or switches them, LOADPLUS leaves the original VSAM data sets intact throughout the load process. This nondestructive
process allows you to easily restart from a failure, or make the objects available without having to recover.

**Tasks to perform before running the load job**

This topic describes what action to take, regarding staging data sets, for LOADPLUS to perform your job correctly.

Table 22: Staging data set actions

<table>
<thead>
<tr>
<th>REDEFINE command or installation option</th>
<th>Object you are loading</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>REDEFINE NO</td>
<td>VCAT-defined</td>
<td>Preallocate the staging data sets and any data sets that you might need for expansion of a multi-data-set object. For information about naming conventions for these data sets, see Table 23 on page 100. If you do not preallocate staging data sets and you specify REDEFINE NO, LOADPLUS fails because it cannot find the data sets.</td>
</tr>
<tr>
<td>STOGROUP-defined</td>
<td></td>
<td>With the exception noted, preallocate the staging data sets and any data sets that you might need for expansion of a multi-data-set object. For information about naming conventions for these data sets, see Table 23 on page 100. If you do not preallocate the staging or expansion data sets, LOADPLUS creates the data sets for you using the same rules as if you had specified REDEFINE YES. <strong>Exception</strong>: Preallocating your staging data sets when you are loading a partition-by-growth table space produces unpredictable results. Allow LOADPLUS to create the data sets for you in this case.</td>
</tr>
<tr>
<td>REDEFINE YES</td>
<td>VCAT-defined</td>
<td>Provide the IDCAMS statements in the SYSIDCIN data set to delete and define the staging data sets. For information about naming conventions for these data sets, see Table 23 on page 100. If you do not provide any statements in SYSIDCIN, LOADPLUS treats the job as if you specified REDEFINE NO.</td>
</tr>
<tr>
<td>STOGROUP-defined</td>
<td></td>
<td>Ensure that you have enough space available for LOADPLUS to automatically allocate the staging data sets.</td>
</tr>
</tbody>
</table>

If you need to preallocate or define the staging data sets, see the section for the process that you are using to determine which naming conventions to use.
Staging data sets and the data set rename process

LOADPLUS uses the rename process when FASTSWITCH NO is in effect. During this process, LOADPLUS replaces the original data sets with the staging data sets by renaming the VSAM data sets.

Naming conventions for the rename process

If you need to preallocate or define the staging data sets, use the original DB2 data set names as a starting point, but replace the DSNDBC and DSNDBD nodes with BMCDBC and BMCDBD. The following table illustrates this naming convention:

Table 23: Naming conventions for staging data sets for the rename process

<table>
<thead>
<tr>
<th>Existing data set name</th>
<th>Staging data set name</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCAT.DSNDBC.database.tableSpace.10001.Annn</td>
<td>VCAT.BMCDBC.database.tableSpace.10001.Annn</td>
</tr>
<tr>
<td>VCAT.DSNDBD.database.tableSpace.10001.Annn</td>
<td>VCAT.BMCDBD.database.tableSpace.10001.Annn</td>
</tr>
</tbody>
</table>

Data set rename process

After writing the loaded objects to the staging data sets (named as shown in Table 23 on page 100), LOADPLUS renames the data sets by using the following process. Figure 5 on page 102 illustrates this process.

Note
You can control when this process occurs by using the SWITCHTIME option. For more information, see “SWITCHTIME” on page 255.

1 LOADPLUS prevents all access to the objects and renames the original DB2 VSAM data sets by replacing DSN in the DSNDBC node of the cluster and in the DSNDBD node of the data component with OLD.

Note
LOADPLUS terminates if the following data sets already exist:
- VCAT.OLDDBC.database.tableSpace
- VCAT.OLDDBD.database.tableSpace

This part of the process makes the original data sets obsolete.
2 LOADPLUS renames the staging data sets by replacing BMC in the BMCDBC node of the cluster and in the BMCDBD node of the data component with DSN. This step gives the staging data sets the same name as the original data sets, even though they are in a new location.

3 After successfully renaming the data sets, LOADPLUS starts the objects with their original statuses.

4 The next step depends on the value for the ORIGDISP or ORIGINALDISP option:

- If the value for the ORIGDISP or ORIGINALDISP option is DELETE, LOADPLUS deletes the now-obsolete original data sets.

- If the value for this option is RENAME, LOADPLUS renames the original data sets to the staging data set names by replacing OLD with BMC. RENAME preserves the space that was initially allocated for the original data sets. As a result, the staging data sets are ready to be used with the next load job that uses staging data sets.
Staging data sets and the FASTSWITCH process

In the FASTSWITCH process, LOADPLUS bypasses the VSAM rename process and changes the DB2 catalog entry to point to the staging data sets.

To use the FASTSWITCH process, you must take one of the following actions:

- Set the FASTSWITCH command or installation option to YES.
- Set the FASTSWITCH installation option to ZPARM, and ensure that the DB2 ZPARM value is YES.

For more information about this option, see “FASTSWITCH” on page 253.
**Naming conventions for the FASTSWITCH process**

If you need to preallocate or define the staging data sets, use the original DB2 data set names as a starting point, but replace the `I` qualifier with `J`, or the `J` qualifier with `I`. The following table illustrates this naming convention:

<table>
<thead>
<tr>
<th>Existing data set name</th>
<th>Staging data set name</th>
</tr>
</thead>
</table>

**Data set FASTSWITCH process**

After writing the loaded objects to the staging data sets (named as shown in Table 24 on page 103), LOADPLUS replaces the original data sets by using the following process. Figure 6 on page 104 illustrates this process.

*Note*

You can control when this process occurs by using the SWITCHTIME option. For more information, see “SWITCHTIME” on page 255.

1. LOADPLUS prevents all access to the objects and updates the DB2 catalog (SYSIBM.SYSTABLEPART and SYSIBM.SYSINDEXPART) to point to the staging data sets.

   This part of the process makes the original data sets obsolete.

2. After the switch is successful and the LOAD or COPY row is registered in the SYSIBM.SYSCOPY catalog table, LOADPLUS starts the objects with their original statuses.

3. The next step depends on the value for the ORIGDISP or ORIGINALDISP option:
   - If the value for the ORIGDISP or ORIGINALDISP option is DELETE, LOADPLUS deletes the now-obsolete original data sets.
   - If the value for this option is RENAME, LOADPLUS retains the staging data sets but does not rename them.

   The data sets still exist with their original names, preserving the space. You can use them as staging data sets in a future load job.
Figure 6: Data set FASTSWITCH process

Tasks to perform if your load job terminates

If you terminate the load job or the job abnormally terminates and you do not intend to restart it, you can delete the staging data sets. However, if the job terminates during the rename process, BMC recommends that you restart your job rather than attempt to rename the staging data sets manually.
Considerations when loading a large number of partitions

For most load jobs, LOADPLUS can load up to 4096 partitions without encountering any problems. However, loading objects with a large number of partitions increases the potential for encountering performance problems and memory restrictions.

Consider the information in the following topics to help you avoid these issues.

Definition of a large number of partitions

The topics in this section discuss considerations and recommendations when loading a large number of partitions.

Whether the number of partitions that you have is considered to be a large number depends on your environment. In the BMC testing environment, the following numbers applied to LOADPLUS:

- For general recommendations described in this book, a large number of partitions was considered to be approximately 500 or more partitions.

- For copy recommendations, a large number of partitions was considered to be a total number of copy data sets (including local and remote copies) of approximately 1500 or more.

Recommended option values for loading a large number of partitions

Changing the following command and installation option values can improve the performance of your job and help avoid memory or other restrictions:

- If you are replacing only a few partitions and all of the remaining partitions are empty, specify LOAD RESUME YES PART n REPLACE instead of LOAD REPLACE.

- Consider specifying REDEFINE NO, either in the installation options or on the LOAD command. This option creates additional data sets only when needed and cleans up unused data sets.

- If you are loading more than 2000 partitions, BMC recommends that you increase the value of your STOPRETRY installation option to a minimum of 450. Increasing
Considerations when loading a large number of partitions

When you are loading a large number of partitions, the potential for encountering constrained resources is greater than for other objects.

Use the following information to avoid or work around this situation:

- If you dynamically allocate partition-level copies, LOADPLUS appends the partition number to the default ddname prefix. If you dynamically allocate partition-level copy data sets for more than 99 partitions, ensure that you change the value of the COPYDDN installation or command option to specify a ddname prefix; the prefix plus the highest partition number must not exceed eight characters. If you are also dynamically allocating remote copy data sets, change the value of the RECOVERYDDN command option (or RCVYDDN installation option) to specify a ddname; the prefix plus the highest partition number must not exceed eight characters.

- Ensure that the value of your LOCKROW installation option is YES.

- Consider using VOLCNT (AUTO,AUTO) instead of the default value of (25,25) to avoid data set allocation limitations.

Recommendations for avoiding constrained resources when loading a large number of partitions

When you are loading a large number of partitions, the potential for encountering constrained resources is greater than for other objects.

Use the following information to avoid or work around this situation:

- If you need to make copies, consider one of the following options to avoid encountering data set allocation restrictions of the operating system or LOADPLUS memory restrictions:
  
  — Limit the number of copies per partition.
  
  — If you are loading all partitions, specify COPYLVL FULL to create a single copy.
  
  — If you are loading a subset of contiguous partitions, create a single copy by specifying COPYSUBSET=YES in the installation options module and, if you are dynamically allocating copy data sets, COPYLVL FULL.

  For more information, see the COPYLVL option on “COPYLVL” on page 299 or “COPYLVL=FULL” on page 719, and “COPYSUBSET=NO” on page 720.

  **Note**
  
  For partition-by-growth table spaces, these allocation and memory restrictions apply only when you are making inline copies and you also specify ORDER NO.

- If you dynamically allocate partition-level copies, LOADPLUS appends the partition number to the default ddname prefix. If you dynamically allocate partition-level copy data sets for more than 99 partitions, ensure that you change the value of the COPYDDN installation or command option to specify a ddname prefix; the prefix plus the highest partition number must not exceed eight characters. If you are also dynamically allocating remote copy data sets, change the value of the RECOVERYDDN command option (or RCVYDDN installation option) to specify a ddname; the prefix plus the highest partition number must not exceed eight characters.

- Ensure that the value of your LOCKROW installation option is YES.

- Consider using VOLCNT (AUTO,AUTO) instead of the default value of (25,25) to avoid data set allocation limitations.
■ Ensure that you have specified a region size that allows the system to allocate as much virtual storage as possible to the LOADPLUS job. BMC recommends that you specify REGION=0M on the JOB or EXEC statement of your execution JCL.

■ If you have specified REGION=0M and your load job fails with a constrained resources error, consider loading fewer partitions in a single job.

■ When you are loading a large number of partitions and dynamic allocation is active, BMC recommends that you specify DYNAMNBR=1600 on your EXEC statement. Including this parameter enables the system to acquire adequate resources immediately before their use and to release them immediately after use.

■ Loading a large number of compressed partitions might result in constrained resources. If you encounter this problem, consider specifying fewer partitions in a single job.

■ If you are running a two-phase load job that fails with a constrained resources error, try restarting the job. The restarted job might have lower memory requirements.

Additional recommendations when loading a large number of partitions

If you are loading a large number of partitions, you might need to increase the size of the following common utility table spaces from the standard size that was allocated during installation:

■ BMCSYNC
  Estimate the allocation for this table space based on all of the following factors:
  — Number of utilities that you are running concurrently
  — Number of partitions that you are processing concurrently
  — Number of files that you are dynamically allocating

■ BMCDICT
  If you are loading compressed data, estimate the allocation for this table space by multiplying by 64 KB the number of compressed partitions that you are operating on concurrently (loading with LOADPLUS and reorganizing with REORG PLUS).
Considerations when loading partition-by-growth table spaces

You can use LOADPLUS to load partition-by-growth table spaces.

If needed, LOADPLUS adds partitions during the load as long as you have not reached the limit specified by MAXPARTITIONS in your table space definition. The following topics describe information to consider when loading partition-by-growth table spaces.

Note

If you are loading partition-by-growth table spaces that contain a LOB column, LOADPLUS invokes DSNUTILB. The information in this section does not apply in this case.

How LOADPLUS handles new partitions

The following considerations apply when LOADPLUS adds new partitions:

- If you are making copies, LOADPLUS copies all partitions, including those that it adds to the table space. Therefore, you must consider possible expansion when allocating your copy data sets. BMC recommends that you enable dynamic allocation for your copy data sets when loading partition-by-growth table spaces. For more information, see “Copy data sets in LOADPLUS” on page 436.

- How LOADPLUS handles compression dictionaries for new partitions depends on the type of job that you are running:
  — For LOAD REPLACE, LOADPLUS uses the dictionary from the first partition.
  — For LOAD RESUME YES, LOADPLUS uses the dictionary from the partition in which the load started adding rows.

Row format considerations for partition-by-growth table spaces

For partition-by-growth table spaces, LOADPLUS uses the existing row format of the table space.

The following requirements apply:

- All partitions in the table space must be in the same row format.
The partitions must be in the following format, based on the DB2 subsystem parameter RRF:

— When RRF=ENABLE, the format must be reordered row format (RRF).
— When RRF=DISABLE, the format must be basic row format (BRF).

**SHRLEVEL considerations for partition-by-growth table spaces**

The following considerations apply to the SHRLEVEL option that you specify:

- For load jobs that use staging data sets, do not preallocate your staging data sets when loading partition-by-growth table spaces. Allow LOADPLUS to handle the allocation.

- For LOAD REPLACE SHRLEVEL CHANGE, LOADPLUS does not support partitions that are added during your load other than those that LOADPLUS adds. If LOADPLUS detects that a new partition is added after the load has started, LOADPLUS issues messages BMC50225E and BMC50010U and terminates with return code 16.

- For a SQLAPPLY load, message BMC51579I does not indicate how many rows were loaded into each partition. Use the information in this message only to ensure that the total number of rows loaded into the table space is correct.

**Other option considerations when loading partition-by-growth table spaces**

The following considerations apply to specific options when you are loading partition-by-growth table spaces:

- You cannot specify the INTO PART option (with or without REPLACE) when loading a partition-by-growth table space.

- If you are making partition-level copies and sending them to tape, ensure that the value for the MAXTAPE option takes potential expansion into account.
Restart limitations when loading partition-by-growth table spaces

The following limitations apply to restarting your load job when you are loading a partition-by-growth table space:

- If the value for MAXPARTITIONS changes between the time the job fails and the time you restart it, LOADPLUS issues message BMC50221U and terminates.

- If DB2 adds partitions before you restart a job, LOADPLUS issues message BMC50221U and terminates.

- If a SHRLEVEL NONE load job fails because insufficient space is available, BMC recommends that you perform an ALTER to decrease the PCTFREE or FREEPAGE value before restarting the job. However, LOADPLUS only honors these changes in the following situations:
  
  — For a single-phase load, LOADPLUS honors these changes for those partitions that it has not started loading.
  
  — For a two-phase load, LOADPLUS honors these changes only if the failure occurs during the PRELOAD phase.

BMCSYNC table considerations when loading partition-by-growth table spaces

When you are loading a partition-by-growth table space, you might need to increase the size of the BMCSYNC table space from the standard size that was allocated during installation.

Estimate this allocation based on all of the following factors:

- The number of utilities that you are running concurrently
- The value of MAXPARTITIONS
- The number of files that you are dynamically allocating

Considerations when loading identity columns

When loading identity columns, LOADPLUS generates values as necessary and updates the DB2 catalog to reflect the maximum assigned value.
How LOADPLUS generates identity column values

When the table that you are loading contains an identity column, LOADPLUS generates identity column values in the following cases:

- You include a field specification on your LOAD command, but you do not include a field specification for the identity column.

- You include a field specification for the identity column but a DEFAULTIF clause on the field specification is true.

For all types of load jobs except SQLAPPLY, LOADPLUS reserves one or more caches of identity column values by updating the MAXASSIGNEDVAL field of SYSIBM.SYSSEQUENCES during the PRELOAD or COMBINED phase. LOADPLUS reserves at least one cache for each read task that the load job starts.

You can use the IDCACHE command or installation option to control the number of values in the cache that LOADPLUS reserves. Specifying a large cache increases the possibility of gaps in the set of identity column values and the risk of running out of identity column values. In addition, if your identity column is defined with CYCLE YES, a large cache size introduces a greater likelihood of generating duplicate values. However, specifying a smaller cache size can impact performance because LOADPLUS must access the DB2 catalog more frequently.

Note
For an SQLAPPLY load job, LOADPLUS does not reserve an identity column cache on the DB2 catalog. For this type of load, LOADPLUS uses DB2 to generate identity column values.

How LOADPLUS updates MAXASSIGNEDVAL when loading identity columns

LOADPLUS updates the value of the MAXASSIGNEDVAL field of SYSIBM.SYSSEQUENCES based on the values that you are loading.
Generated values

When generating your identity column values, LOADPLUS always updates the MAXASSIGNEDVAL field, regardless of the value of the UPDATEMAXA command option or UPDMAXA installation option. LOADPLUS updates this field at the time that it reserves each cache and updates it with the last value in the cache that it is reserving.

Input file values

When loading your identity column values from an input file, LOADPLUS updates MAXASSIGNEDVAL only if you specify YES for the UPDATEMAXA command option or the UPDMAXA installation option. As the following table describes, the value with which LOADPLUS updates this field depends on the identity column values loaded.

LOADPLUS resets the cache when updating the MAXASSIGNEDVAL field.

Table 25: How LOADPLUS updates MAXASSIGNEDVAL when loading values from an input file

<table>
<thead>
<tr>
<th>Loaded identity column values</th>
<th>LOADPLUS action on MAXASSIGNEDVAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>The maximum loaded identity column value is</td>
<td>LOADPLUS updates MAXASSIGNEDVAL with the maximum loaded identity column value.</td>
</tr>
<tr>
<td>■ Greater than the value of MAXASSIGNEDVAL (for positive INCREMENT values; less than the value of MAXASSIGNEDVAL for negative INCREMENT values) and ■ Within the range that is defined on the identity column</td>
<td></td>
</tr>
</tbody>
</table>

| The maximum loaded identity column value is less than the value of MAXASSIGNEDVAL (for positive INCREMENT values; greater than MAXASSIGNEDVAL for negative INCREMENT values). | LOADPLUS does not update MAXASSIGNEDVAL. |

| The maximum loaded identity column value is not within the range that is defined on the identity column, but the minimum loaded identity column value is within this range. | LOADPLUS updates MAXASSIGNEDVAL with the highest value in the range defined on the identity column (for positive INCREMENT values). If INCREMENT is a negative value, LOADPLUS updates MAXASSIGNEDVAL with the lowest value in the range defined on the identity column. |

| The maximum and minimum loaded identity column values are not within the range that is defined on the identity column. | LOADPLUS does not update MAXASSIGNEDVAL. |
How LOADPLUS handles invalid value errors when generating identity column values

When generating identity column values, LOADPLUS validates that these values are within the range that is defined on the column.

LOADPLUS handles any invalid identity column values based on the type of load job and the value for the IDERROR command or installation option.

- For SQLAPPLY load jobs, LOADPLUS discards the rows with the invalid values and ignores the value for the IDERROR option.

- For all other load jobs:
  - If you specify a value of DISCARD for IDERROR, LOADPLUS discards the rows with the invalid values.
  - If you specify a value of FAIL for IDERROR, LOADPLUS terminates the load job as soon as an identity column value is generated that exceeds the range that is defined on the column.

Additional identity column considerations

This topic describes additional considerations for loading identity columns.

Authorization

To load data into tables that contain identity columns, you must have additional authorization as described in “Setting LOADPLUS authorizations” on page 51.

FORMAT BMCUNLOAD

If the identity column in the table that you are loading is defined as GENERATED ALWAYS, you must specify IDENTITYOVERRIDE YES.

DB2 check constraints

When an identity column either contains a check constraint or is referenced in the check condition of a check constraint, and LOADPLUS would generate values for that identity column, LOADPLUS terminates.
**Order of identity column values**

When multitasking your load job, you cannot ensure that LOADPLUS will assign generated identity column values in input order. To force LOADPLUS to assign identity column values in input order, build your load job in one of the following ways. If you do not use one of the following specifications, the order in which the generated values are assigned is unpredictable.

- For an SQLAPPLY load, specify APMAXAGENTS 1 and ORDER NO.

  **Note**
  
  For an SQLAPPLY load, you can also have values assigned in clustering or partitioning key order. To do so, specify APMAXAGENTS 1 and ORDER YES.

- For all other load types, specify only one SYSREC data set per load job.

**Field specifications**

LOADPLUS terminates when both of the following conditions exist:

- An identity column is referenced in a WHEN, DEFAULTIF, or NULLIF statement on the field specification of another column.

- LOADPLUS would generate values for that identity column.

Additionally, LOADPLUS terminates when both of the following conditions exist:

- You include a field specification for an identity column that is defined as GENERATED ALWAYS.

- You do not specify IDENTITYOVERRIDE.

**XML data**

LOADPLUS loads XML data either natively or by invoking DSNUTILB, depending on the type of load that you are running and the object that you are loading.

**Loading XML data natively**

LOADPLUS loads XML data natively in most cases.
For those conditions under which LOADPLUS invokes DSNUTILB, see “Loading XML data by invoking DSNUTILB” on page 118.

**To load XML data natively**

1. Create your load job as usual, adjusting it if necessary based on the information described in “Considerations for loading XML data natively” on page 115.

2. Use one of the following options for your field specification:
   - If your input file contains the XML data, either do not include a field specification for this job, or specify the XML data type in your field specification for the XML column.
   - If your input file references a BLOB or CLOB file that contains your XML data, specify the BLOBF or CLOBF subtype, respectively, on the appropriate CHAR or VARCHAR data type. Specify any applicable subtype options.

3. If you are loading from an input data set that is in spanned-record format:
   - Ensure that the XML (and any LOB) fields follow all other fields in your field specification.
   - Specify FORMAT SPANNED YES.
   
     For more information about this option, including restrictions, see “SPANNED” on page 201

4. If you specify COPY YES, ensure that dynamic allocation is enabled for your XML copy data sets.

5. *(optional)* Adjust the value for the XMLAVGSIZE option on your LOAD command or the AVGSIZE option on your field specification.

   For more information about these options, see “XMLAVGSIZE” on page 288 or “AVGSIZE” on page 355.

6. Ensure that the SMF MEMLIMIT parameter meets the minimum requirements for loading your XML data.

   For more information, see “Overriding the MEMLIMIT system parameter” on page 55.

**Considerations for loading XML data natively**

The following considerations apply when loading XML data natively.
**Requirements when using referenced files**

When your input data set contains a reference to the file that contains your XML data, the following requirements apply:

- If your referenced file is a partitioned data set (PDS) or extended partitioned data set (PDSE), you must include a member name in the name that is provided in the input file.

- If your referenced file is a hierarchical file system (HFS) file, you must preallocate and mount the file system before running the load job.

The input file must specify the full path name of the file, as in the following example:

```
/home/rdajdm/lobfr1
```

**Copy data set requirements**

If you specify COPY YES, you must enable dynamic allocation for at least the primary local XML copy data set (DDTYPE LOCPXCPY). In general, LOADPLUS ignores any XML copy data set allocation in your JCL.

LOADPLUS allocates your XML copy data sets by using the following naming convention. If you include a DD statement in your JCL that uses the nonnumeric portion of this naming convention as the ddname (or prefix), LOADPLUS terminates.

```
X[L|R][P|B]nnnnn
```

These variables indicate the following information:

- [L | R] is either L for local copy or R for remote copy.
- [P | B] is either P for primary copy or B for backup copy.
- nnnnn is a number that LOADPLUS generates for that particular copy data set.

For example, XLP00001 indicates a local primary copy.

**Incompatible options**

The following table describes restrictions that apply to LOADPLUS options when loading XML data.

Additionally, LOADPLUS terminates in either of the following cases:

- You attempt to load a base table that contains an XML column and you specify LOAD RESUME YES PART n REPLACE.
- You specify an XML column on any predicate of a WHEN statement.

### Table 26: Options that are incompatible with an XML load

<table>
<thead>
<tr>
<th>Option</th>
<th>LOADPLUS response</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULTIF</td>
<td>Terminates if specified on an XML column</td>
</tr>
<tr>
<td>FORMAT BMC</td>
<td>Terminates</td>
</tr>
<tr>
<td>FORMAT BMCUNLOAD</td>
<td>Terminates</td>
</tr>
<tr>
<td>FORMAT UNLOAD</td>
<td>Terminates</td>
</tr>
<tr>
<td>FORMAT CSV</td>
<td>Terminates</td>
</tr>
<tr>
<td>INDEX UPDATE</td>
<td>Terminates</td>
</tr>
<tr>
<td>INLINE YES</td>
<td>Changes to INLINE NO</td>
</tr>
<tr>
<td>PRELOAD ANALYZE</td>
<td>Terminates</td>
</tr>
<tr>
<td>PRELOAD PAUSE</td>
<td>Terminates</td>
</tr>
<tr>
<td>SKIPIX</td>
<td>Ignores this option for the following types of indexes:</td>
</tr>
<tr>
<td></td>
<td>- Document ID</td>
</tr>
<tr>
<td></td>
<td>- Node ID</td>
</tr>
<tr>
<td>VALUE</td>
<td>Terminates if specified on an XML column</td>
</tr>
</tbody>
</table>

### Additional considerations

The following additional considerations apply when loading XML data:

- If the XML table space and index spaces are defined with DEFINE NO, LOADPLUS materializes these data sets only if the XML column in the base table is participating in the load.

- If you specify LOAD RESUME YES TABLE `tableName` REPLACE, LOADPLUS loads the base table as usual. The XML table space for that table is loaded as if you specified LOAD REPLACE.

- When the following conditions exist, LOADPLUS generates XML copies at the partition level regardless of any other options, such as COPYLVL, that you specify:
  - You specify LOAD RESUME YES (without PART REPLACE) COPY YES.
  - Your XML table space is partition-by-growth.
LOADPLUS generates partition-level copies for only the partitions that you are loading.
LOADPLUS generates base table copies as usual, honoring other copy options such as COPYLVL.

- Because the document ID (DOCID) column of the base table is an identity column, the considerations described in “Considerations when loading identity columns” on page 110 apply.

- If the XML table space is a partition-by-growth table space, you might need to increase the size of your BMCSYNC table. 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

For XML table spaces that are partition-by-growth table spaces, the considerations described in “Considerations when loading partition-by-growth table spaces” on page 108 apply.

### Loading XML data by invoking DSNUTILB

LOADPLUS invokes DSNUTILB for the following XML load jobs:

- An SQLAPPLY load
- A two-phase load (PRELOAD CONTINUE)
- When loading XML data from a DBCLOB referenced file
- When loading an XML column that is defined with an XML schema
- When loading a table containing a user-defined XML index that is defined with DEFINE NO and is not yet materialized

For considerations when LOADPLUS invokes DSNUTILB, see “Load jobs that invoke DSNUTILB” on page 57

#### To load XML data when LOADPLUS invokes DSNUTILB

1. Ensure that you meet the requirements for running a load that invokes DSNUTILB.
   
   See “Enabling LOADPLUS to invoke DSNUTILB” on page 58.

2. If you are loading from an input data set that is in spanned-record format:
   
   a. Ensure that the XML (and any LOB) fields follow all other fields in your field specification.
b Specify FORMAT SPANNED YES.

For more information about this option, including restrictions, see “SPANNED” on page 201

3 Use one of the following options for your field specification:

- If your input file contains the XML data, either specify the XML data type in your field specification for the XML column, or do not include a field specification for this job.

- If your input file references the file that contains your XML data, specify the appropriate CHAR or VARCHAR subtype (BLOBF, CLOBF, or DBCLOBF).

## LOB data

LOADPLUS loads LOB data, including inline LOB data, either natively or by invoking DSNUTILB, depending on the type of load that you are running and the object that you are loading.

### Loading LOB data natively

LOADPLUS loads LOB data natively in most cases.

For the conditions under which LOADPLUS invokes DSNUTILB, see “Loading LOB data by invoking DSNUTILB” on page 122.

**To load LOB data natively**

1 Create your load job as usual, adjusting it if necessary based on the information described in “Considerations for loading LOB data natively” on page 120.

2 Use one of the following options for your field specification:

- If your input file contains the LOB data, either do not include a field specification for this job, or specify a LOB data type (BLOB, CLOB, or DBCLOB) in your field specification for the LOB column.

- If your input file references the file that contains your LOB data, specify the appropriate CHAR or VARCHAR subtype (BLOBF, CLOBF, or DBCLOBF).

3 If you are loading from an input data set that is in spanned-record format:
Considerations for loading LOB data natively

The following considerations apply when loading LOB data natively.

Requirements when using referenced files

If your input data set contains a reference to the file that contains your LOB data, note the following requirements:

- If your referenced file is a PDS or PDSE, you must include a member name in the name provided in the input file.

- If your referenced file is an HFS file, you must preallocate and mount the file system before running the load job.

Also, your input file must specify the full path name of the file, as in the following example:

/home/rdajdm/lobfr1
Copy data set requirements

If you specify COPY YES, you must enable dynamic allocation for at least the primary local LOB copy data set (DDTYPE LOCPLCPY). In general, LOADPLUS ignores any LOB copy data set allocation in your JCL.

LOADPLUS allocates your LOB copy data sets by using the following naming convention. If you include a DD statement in your JCL that uses the nonnumeric portion of this naming convention as the ddname (or prefix), LOADPLUS terminates.

L[L|R][P|B]nnnnn

These variables indicate the following information:

- [L|R] is either L for local copy or R for remote copy.
- [P|B] is either P for primary copy or B for backup copy.
- nnnnn is a number that LOADPLUS generates for that particular copy data set.

For example, LLP00002 indicates a local primary copy.

Incompatible options

The following table describes restrictions that apply to options when loading LOB data.

Additionally, LOADPLUS terminates if you specify a LOB column on any predicate of a WHEN statement.

**Table 27: Options that are incompatible with a LOB load**

<table>
<thead>
<tr>
<th>Option</th>
<th>LOADPLUS response</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULTIF</td>
<td>Terminates if specified on a LOB column</td>
</tr>
<tr>
<td>FORMAT BMC</td>
<td>Terminates</td>
</tr>
<tr>
<td>FORMAT BMCUNLOAD</td>
<td></td>
</tr>
<tr>
<td>FORMAT UNLOAD</td>
<td></td>
</tr>
<tr>
<td>FORMAT CSV</td>
<td></td>
</tr>
<tr>
<td>INDEX UPDATE</td>
<td>Terminates</td>
</tr>
<tr>
<td>INLINE YES</td>
<td>Changes this option to INLINE NO</td>
</tr>
<tr>
<td>PRELOAD ANALYZE</td>
<td>Terminates</td>
</tr>
<tr>
<td>PRELOAD PAUSE</td>
<td></td>
</tr>
<tr>
<td>SKIPIX</td>
<td>Ignores this option for row ID or auxiliary indexes</td>
</tr>
</tbody>
</table>
### Additional considerations

The following additional considerations apply when loading LOB data:

- If the LOB table space and index space are defined with DEFINE NO, LOADPLUS materializes these data sets only if the LOB column in the base table is participating in the load. For partitioned base table spaces, LOADPLUS materializes LOB spaces only for the partitions that are participating in the load.

- If you specify LOAD RESUME YES TABLE `tableName` REPLACE, LOADPLUS loads the base table as usual. The LOB table space for this table is loaded as if you specified LOAD REPLACE.

- You might need to increase the size of your BMCSYNC table. 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

### Loading LOB data by invoking DSNUTILB

LOADPLUS invokes DSNUTILB for the following types of LOB load jobs:

- An SQLAPPLY load
- A two-phase load (PRELOAD CONTINUE)
- When loading a partition-by-growth table space
- When loading from a referenced file of a different type than the column type (for example, from a BLOB referenced file to a CLOB column)

For considerations when invoking DSNUTILB, see “Load jobs that invoke DSNUTILB” on page 57

#### To load LOB data when LOADPLUS invokes DSNUTILB

1. Ensure that you meet the requirements for running a load job that invokes DSNUTILB.
2 Include LOADLOBDATA YES on your LOAD command.

3 If you are loading from an input data set that is in spanned-record format:
   a Ensure that the LOB (and any XML) fields follow all other fields in your field specification.
   b Specify FORMAT SPANNED YES.

   For more information about this option, including restrictions, see “SPANNED” on page 201

4 Use one of the following options for your field specification:
   ■ If your input file contains the LOB data, either do not include a field specification for this job, or specify a LOB data type (BLOB, CLOB, or DBCLOB) in your field specification for the LOB column.
   ■ If your input file references the file that contains your LOB data, specify the appropriate CHAR or VARCHAR subtype (BLOBF, CLOBF, or DBCLOBF).

**Unicode data**

The following topics provide instructions and considerations for loading Unicode data.

**Loading Unicode data**

How LOADPLUS loads Unicode data depends on the type of load that you are running and any translation that you are performing during the load.

**Before you begin**

Using the information in “Data translation” on page 383, determine whether the type of load that you are running will invoke DSNUTILB. If LOADPLUS will invoke DSNUTILB, continue with the following procedure. Otherwise, create your job as usual, adjusting it if necessary based on the information in “Considerations when loading Unicode data” on page 124.
To load Unicode data when LOADPLUS will invoke DSNUTILB

1 Ensure that you meet the requirements for running a load job that invokes DSNUTILB.

See “Enabling LOADPLUS to invoke DSNUTILB” on page 58.

2 Ensure that the value of the UNICODE installation option is YES.

This option tells LOADPLUS to invoke DSNUTILB if required when loading Unicode data. It does not set the default for the UNICODE command option.

3 Make any necessary changes to your job based on information in “Considerations when loading Unicode data” on page 124.

4 Specify the UNICODE or CCSID command option as needed.

For information about these options, see “UNICODE” on page 204 and “CCSID” on page 204.

Considerations when loading Unicode data

Consider the following information when using LOADPLUS to load Unicode data.

Data translation

LOADPLUS either loads Unicode data directly or invokes DSNUTILB to load Unicode data, depending on the type of load that you are running and any translation that you are performing. For more information, see “Data translation” on page 383.

For translations in which LOADPLUS invokes DSNUTILB, see “INTO TABLE statement for a DSNUTILB load” on page 61 for compatibility of the INTO statement that you specify and any data type conversion that you request.

Recommended option specifications

BMC recommends specifying the following options when loading Unicode data:

- ASCII, EBCDIC, or UNICODE
  Specify the appropriate keyword for your input data.
- **CCSID**
  Ensure that you use the encoding scheme for which the input data was created, and that the encoding scheme matches the ASCII, EBCDIC, or UNICODE keyword that you specify. For more information, see “CCSID” on page 204.

- **NOSUBS**
  Specifying this option ensures that LOADPLUS does not change your data when it encounters characters that require substitution. For more information, see “NOSUBS” on page 205.

- **TRIM**
  BMC strongly recommends that you specify this option for CHAR and VARCHAR fields if you do not need to preserve trailing blanks. If translation causes an expansion of the data, specifying this option prevents LOADPLUS from discarding records that you do not want to discard. For more information, see “CHAR” on page 338 or “VARCHAR” on page 340.

### Constants

When loading Unicode data, LOADPLUS handles constants in your LOAD command based on your command and installation options. If you specify a hexadecimal string for the constant, the string must be in the internal format of either the input or the output, depending on the option. The following sections describe these requirements.

LOADPLUS does not translate hexadecimal strings. LOADPLUS does not support the hexadecimal graphic string constants UX'xxxx' and GX'xxxx'.

The following table describes constant considerations by option:

<table>
<thead>
<tr>
<th>Option</th>
<th>Considerations</th>
</tr>
</thead>
</table>
| WHEN, NULLIF, and DEFAULTIF | You can refer to Unicode columns in the predicate of a WHEN, NULLIF, or DEFAULTIF option. LOADPLUS assumes that the constant that you specify is in EBCDIC format and translates it according to the value of the RULES option:  
  - If RULES=BMC, LOADPLUS translates the constant to the CCSID of the table space. If you specify a hexadecimal string for the constant, the string must be in the internal format of the output.  
  - If RULES=STANDARD, LOADPLUS translates the constant to the CCSID of the input data. If you specify a hexadecimal string for the constant, the string must be in the internal format of the input. |
<table>
<thead>
<tr>
<th>Option</th>
<th>Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>When you are loading Unicode data, LOADPLUS assumes that the VALUE constant that you specify is in EBCDIC format, and translates it to the CCSID of the table space. If you specify a hexadecimal string for the constant, the string must be in the internal format of the table space.</td>
</tr>
</tbody>
</table>
| FORMAT CSV| When your input data is Unicode, LOADPLUS assumes that the values for the TERMINATEDBY, ENCLOSEDBY, and AND options are in EBCDIC format and translates them, if necessary, to the CCSID of the input data. Note the following restrictions for these options:  
  ■ Do not specify values for these options that would expand to multiple-byte characters when they are translated.  
  ■ If you specify a hexadecimal value, it must be in the internal format of the input and the length must be no more than one byte. |
| CONTINUEIF| When loading Unicode data, LOADPLUS assumes that the CONTINUEIF constant that you specify is in EBCDIC format, and translates it, if necessary, to the CCSID of the input data. If you specify a hexadecimal value, it must be in the internal format of the input. |

**TRUNCATE option**

When loading Unicode data, LOADPLUS performs truncation after translation.

**CSV data**

LOADPLUS enables you to load data from comma-separated-value (CSV) files. (CSV files are also referred to as comma-delimited or delimited files.)

**Loading CSV data**

Use this procedure to load data from CSV files.

**To load CSV data**

1. Ensure that your data is formatted within the requirements that are specified in “CSV data format” on page 127.

2. Specify FORMAT CSV with your LOAD command.
Depending on the format of your CSV file, you might also need to specify one or more CSV options. For more information about this option, see “CSV” on page 199.

3 Include appropriate field specifications with your LOAD command.

For considerations regarding field specifications, see “CSV field specifications” on page 128.

**CSV data format**

Ensure that your CSV data meets the following requirements:

- Your CSV file must contain delimiters. *Enclosure characters* (characters that enclose each field) in your CSV file are optional.

- Input strings should not contain the same characters as your field delimiters or enclosure characters. If they do, LOADPLUS might load the record incorrectly or reject it. To prevent this problem, complete one of the following tasks:
  - Remove these characters from your input strings.
  - Change the delimiter or enclosure character in your input file so that you do not have a conflict with embedded characters. Then, ensure that the TERMINATEDBY, ENCLOSEDBY, and AND parameters on your FORMAT CSV option are compatible with the delimiter and enclosure characters in your input file.

- Your CSV input file should use a variable-length record format (RECFM=VB). If your input file uses a fixed-length record format, and the last field on a record does not extend to the end of that record, the field must end with a delimiter.

- A one-to-one correspondence should exist between the input fields of your CSV data and the field specifications on your LOAD command. Otherwise, the following results occur. (For more information, see “CSV field specifications” on page 128.)
  - If you have more input fields than field specifications, LOADPLUS ignores the extra input file data. LOADPLUS maps field specifications to input fields in order, starting with the first specification and the first input field.
  - If you have fewer input fields than field specifications, LOADPLUS ignores the additional field specifications.

- LOADPLUS supports loading data that was created by using the FORMAT CSV option of UNLOAD PLUS. However, when loading CSV data, LOADPLUS requires that DATE, TIME, and TIMESTAMP data be in DB2 EXTERNAL format.
The control cards that UNLOAD PLUS produces do not specify an external format for these data types. Either modify these control cards, or load the data by using a LOADPLUS job that provides the correct external format for these data types.

---

**Note**

Although supported, using the FORMAT CSV option of LOADPLUS and UNLOAD PLUS is not the most efficient method for using these two products to migrate DB2 data.

If your table structures are almost identical, BMC recommends that you specify FORMAT BMCLOAD (in UNLOAD PLUS) and FORMAT BMCUNLOAD (in LOADPLUS) to migrate DB2 data. For more information, see “Using data from UNLOAD PLUS in internal BMC format” on page 132.

If your table structures are not similar enough to use this method, BMC recommends that you use the FORMAT INTERNAL option of UNLOAD PLUS. For more information, see the *UNLOAD PLUS for DB2 Reference Manual*.

---

**Installation option considerations when loading CSV data**

When you specify FORMAT CSV, LOADPLUS changes the RULES installation option to STANDARD, regardless of the value specified at installation.

**FORMAT CSV restrictions**

LOADPLUS terminates when either of the following conditions exists and you specify FORMAT CSV:

- You are loading LOB or XML data.
- You also specify CONTINUEIF.

**CSV field specifications**

You must include a field specification on your LOAD command for every column that you are loading.

In addition, each field specification must map to an input field in your CSV file. LOADPLUS maps field specifications to input fields in order, starting with the first specification and the first input field.
**Note**

Alternatively, you can elect not to include any field specifications. If you do not include any field specifications, LOADPLUS loads the data as if you had included a field specification that maps to every column in the table that you are loading.

**Excluding particular input fields**

If you do not want to load data from a particular input field, complete one of the following tasks:

- If the input field that you do not want to load is the last field (or fields) in the CSV file, you do not need to include a field specification for that field. LOADPLUS ignores any extra fields at the end of the input file.

- If the input field that you do not want to load is not the last field (or fields) in the CSV file, include a placeholder field specification on your LOAD command. In this field specification, use a field name that is not the name of a column in the DB2 table. You must also specify SKIPFIELDS YES on your LOAD command.

Figure 7 on page 131 illustrates how LOADPLUS maps your field specifications for CSV files to your DB2 table columns. Example 2a does not supply a field specification for the employee department field. Therefore, LOADPLUS loads employee department data into the employee number column, which is not the desired result. Example 2b corrects this problem by adding a placeholder field specification.

**Field specification requirements**

Your field specifications must meet the following requirements:

- For fields that you load, field names must match the corresponding column names.

- For DATE, TIME, or TIMESTAMP data, your field specification must be an EXTERNAL data type. For DATE and TIME data, you must specify one of the delimited formats listed in the tables in “Supported external DATE TIME and TIMESTAMP formats” on page 356. For delimited TIMESTAMP data, if you do not specify a delimited format, LOADPLUS defaults to format D2E.

- To have LOADPLUS interpret two consecutive delimiter characters as a null field, include the following NULLIF option:

  ```
  NULLIF fieldName = ''
  ```

  If you do not include the NULLIF option, LOADPLUS treats input fields denoted by two consecutive delimiter characters in the following ways:
— For DATE, TIME, and TIMESTAMP fields where the column is nullable, LOADPLUS loads a null value.

— For DATE, TIME, and TIMESTAMP fields where the column is defined as NOT NULL, LOADPLUS loads the default date, time, or timestamp.

— LOADPLUS treats all other fields as VARCHAR fields with length 0.

**Additional field specification considerations**

Consider the following additional information about field specification options:

- You can use NULLIF and DEFAULTIF field specifications. However, be aware that LOADPLUS changes the value of the RULES installation option to STANDARD, affecting how LOADPLUS handles NULLIF and DEFAULTIF specifications.

- LOADPLUS ignores the following field specifications:
  
  — POSITION (start:end)
    
    LOADPLUS starts from the first position of each field. However, BMC recommends that you specify POSITION(*).

  — Data type, except on specifications for DATE, TIME, and TIMESTAMP fields
    LOADPLUS performs data type conversion only to DATE, TIME, and TIMESTAMP external types.

  — Length

**Example field mapping**

The following figure illustrates how LOADPLUS maps your field specification to the table that you are loading:
Figure 7: CSV field specification mapping

**CSV file**

"Sue Lee", "03/10/1994", "901", "12634", ",
"Jane Doe", "04/16/1996", "902", "32108", ",
"Juan Lopez", "12/12/1993", "900", "23145", ,

**DDL for EMPLOYEE1**

```sql
CREATE TABLE EMPLOYEE1
    (EMP_NAME CHAR(30)
    ,EMP_DATE DATE
    ,EMP_DEPT SMALLINT
    ,EMP_NO SMALLINT
    ,EMP_SALES INTEGER
    )
```

**DB2 table EMPLOYEE1 – example 1**

<table>
<thead>
<tr>
<th>EMP_NAME</th>
<th>EMP_DATE</th>
<th>EMP_DEPT</th>
<th>EMP_NO</th>
<th>EMP_SALES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sue Lee</td>
<td>03/10/1994</td>
<td>901</td>
<td>12634</td>
<td></td>
</tr>
<tr>
<td>Jane Doe</td>
<td>04/16/1996</td>
<td>902</td>
<td>32108</td>
<td></td>
</tr>
<tr>
<td>Juan Lopez</td>
<td>12/12/1993</td>
<td>900</td>
<td>23145</td>
<td></td>
</tr>
</tbody>
</table>

**Field specifications**

```sql
INTO TABLE EMPLOYEE1
    (EMP_NAME POSITION (*)
    ,EMP_DATE POSITION (*) DATE-D1E EXTERNAL
    ,EMP_DEPT POSITION (*)
    ,EMP_NO POSITION (*)
    ,EMP_SALES POSITION (*)
        NULLIF EMP_SALES = ''
    )
```

**DDL for EMPLOYEE2**

```sql
CREATE TABLE EMPLOYEE2
    (EMP_NAME CHAR(30)
    ,EMP_DATE DATE
    ,EMP_DEPT SMALLINT
    ,EMP_NO SMALLINT
    ,EMP_SALES INTEGER
    )
```

**DB2 table EMPLOYEE2 – example 2a – bad results**

<table>
<thead>
<tr>
<th>EMP_NAME</th>
<th>EMP_DATE</th>
<th>EMP_NO</th>
<th>EMP_SALES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sue Lee</td>
<td>03/10/1994</td>
<td>901</td>
<td>12634</td>
</tr>
<tr>
<td>Jane Doe</td>
<td>04/16/1996</td>
<td>902</td>
<td>32108</td>
</tr>
<tr>
<td>Juan Lopez</td>
<td>12/12/1993</td>
<td>900</td>
<td>23145</td>
</tr>
</tbody>
</table>

**DB2 table EMPLOYEE2 – example 2b – good results**

<table>
<thead>
<tr>
<th>EMP_NAME</th>
<th>EMP_DATE</th>
<th>EMP_NO</th>
<th>EMP_SALES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sue Lee</td>
<td>03/10/1994</td>
<td></td>
<td>12634</td>
</tr>
<tr>
<td>Jane Doe</td>
<td>04/16/1996</td>
<td></td>
<td>32108</td>
</tr>
<tr>
<td>Juan Lopez</td>
<td>12/12/1993</td>
<td></td>
<td>23145</td>
</tr>
</tbody>
</table>

**Field specifications**

```sql
INTO TABLE EMPLOYEE2
    (EMP_NAME POSITION (*)
    ,EMP_DATE POSITION (*) DATE-D1E EXTERNAL
    ,EMP_DEPT POSITION (*)
    ,EMP_NO POSITION (*)
    ,EMP_SALES POSITION (*)
        NULLIF EMP_SALES = ''
    )
```
Data and control cards from UNLOAD PLUS

You can use the combination of UNLOAD PLUS and LOADPLUS to migrate data from one table to another.

BMC provides two high-performance migration methods, migrating the data in internal BMC format or in a standard DB2 format:

- To migrate data in internal BMC format (which is the most efficient method and provides the best performance), specify FORMAT BMCLOAD in UNLOAD PLUS to unload the data, and FORMAT BMCUNLOAD in LOADPLUS to reload the data.

- To migrate data in a standard DB2 format, use one of the other FORMAT options in UNLOAD PLUS (such as FORMAT INTERNAL) to unload the data. Then reload the data with LOADPLUS, using the control cards that UNLOAD PLUS generated.

Each method has certain requirements and considerations. For more information, see the following topics:

- “Using data from UNLOAD PLUS in internal BMC format” on page 132
- “Using control cards from UNLOAD PLUS after migrating data in a standard DB2 format” on page 138

Using data from UNLOAD PLUS in internal BMC format

When used in conjunction with UNLOAD PLUS, LOADPLUS provides a high-speed option to move data from one table to another table that has a similar table structure.

This feature is useful, for example, for migrating data from test to development databases. The FORMAT BMCLOAD option of UNLOAD PLUS unloads the data in an internal format that only LOADPLUS can read. Using the FORMAT BMCUNLOAD option of LOADPLUS to read this data significantly improves performance for this type of load job by reducing the need for data verification and conversion. See the UNLOAD PLUS for DB2 Reference Manual for information about unloading the data for this type of data migration.

Loading data with FORMAT BMCUNLOAD

Use this procedure to load data that was unloaded by using the FORMAT BMCLOAD option of UNLOAD PLUS.
For an example that shows you how to use this feature, see “Example 14 LOAD REPLACE from UNLOAD PLUS data in internal format” on page 588.

To load data with FORMAT BMCUNLOAD

1. Ensure that the table that you are loading meets the requirements specified in “Table structure considerations for FORMAT BMCUNLOAD” on page 134.

2. Specify FORMAT BMCUNLOAD with your LOAD command.

3. Include a WHEN TABLE=obid clause with your INTO option.

For more information about this option, see “WHEN” on page 313.

**WARNING**

BMC recommends that you specify input files from a single UNLOAD PLUS step in a single LOADPLUS step. Otherwise, you might encounter data corruption or other unpredictable results.

4. Specify IDENTITYOVERRIDE YES when both of the following conditions exist:
   - You are loading an identity column that is defined as GENERATED ALWAYS.
   - You did not already include IDENTITYOVERRIDE YES as a string in your UNLOAD PLUS CNTLCARDS BMCLOAD option.

5. Specify PERIODOVERRIDE and TRANSIDOVERRIDE on your INTO option when both of the following conditions exist:
   - You are loading a system-period temporal table
   - You did not already include PERIODOVERRIDE and TRANSIDOVERRIDE as a string on the INTO option in UNLOAD PLUS.

6. If you are loading a system-period temporal table, add the appropriate field specifications for the temporal-specific field.

7. If you do not use the LOAD control statements that UNLOAD PLUS generated, ensure that your LOAD command does not contain field specifications or any of the incompatible command options that are described in “Command options that are incompatible with FORMAT BMCUNLOAD” on page 136.

**Note**

When you use this feature, BMC strongly recommends that you use the LOAD control statements that UNLOAD PLUS generates. Using these control statements ensures that your command options comply with the requirements for using this feature.
Table structure considerations for FORMAT BMCUNLOAD

In general, when you use the FORMAT BMCUNLOAD option, the structure of the table that you are loading must be identical to the structure of the table that you unloaded with UNLOAD PLUS.

This section describes considerations regarding the structure of the two tables. For information about how LOADPLUS handles discrepancies in table structure, see “Validation considerations for FORMAT BMCUNLOAD” on page 135.

General structure considerations

The following characteristics of the table structure do not have to be the same on the target as they are on the source:

- Type of table space (for example segmented)
- Indexes
- Index and table space versions

The following specific characteristics must match:

- Coded character set identifier (CCSID)
- Row format (basic or reordered)

Note
For partitioned table spaces, all partitions must be in the same row format. If the partitions are not all in the same row format, you can run a reorganization to convert all partitions to reordered row format. You can run this reorganization on either version 9 or later of the IBM DB2 REORG utility or any supported version of the BMC REORG PLUS product.

Column definitions

The following considerations apply to column definitions in the two tables:

- The number, order, and data type of the columns in the two tables must be the same, but column names can be different.

- If identity column definitions are not the same (for example, the values for START and INCREMENT), LOADPLUS loads the data. If the value of the UPDATEMAXA or UPDMAXA option is YES, LOADPLUS also issues a warning message.

- For VARCHAR, VARBINARY, and VARGRAPHIC columns, target table columns can have a length that is greater than the source table columns. For row ID columns, LOADPLUS does not verify the column length.
Target row ID and row change timestamp columns must be defined as GENERATED BY DEFAULT. If your row ID or row change timestamp column is defined as GENERATED ALWAYS, LOADPLUS discards all rows for that table and issues message BMC51590E.

**Note**
If you are running LOADPLUS in a CHANGE MANAGER worklist environment, see the CHANGE MANAGER documentation for more information about loading row ID columns.

**DB2 user exits**

The following considerations apply to user exits that are defined for the two tables:

- If the source has an EDITPROC, the target must have an EDITPROC with the same name.
- If the source has a VALIDPROC or FIELDPROC, the target must have one, but the names can differ. LOADPLUS displays a warning but loads the rows.

**Validation considerations for FORMAT BMCUNLOAD**

LOADPLUS validates that your table structure is the same by comparing the verification records of the input file that UNLOAD PLUS creates with the structure of the table that you are loading.

If conflicts exist between the verification records and the table that you are loading, LOADPLUS issues the appropriate verification messages as described in this section. Depending on the type of conflict, LOADPLUS takes one of the following actions:

- The load job fails.
- LOADPLUS loads the rows with a warning.
- LOADPLUS discards the rows for the table until it encounters an accurate set of verification records for the same table.

LOADPLUS issues the following messages when validating the table structure:

- BMC51590E or W displays the type of error.
- BMC51591E or W displays the discrepancy between the two tables.
- BMC51592I displays the number of verification records that were read.

If the input file does not contain any verification records, LOADPLUS issues message BMC51590E and ends with return code 16. If the input file contains verification records but none of them match any of the tables that you are loading, LOADPLUS discards all rows, issues message BMC51590E, and ends with return code 0.
Command options that are incompatible with FORMAT BMCUNLOAD

The following command options are not valid with FORMAT BMCUNLOAD:

- You cannot include a field specification when you specify FORMAT BMCUNLOAD. If your LOAD command includes a field specification, LOADPLUS issues message BMC51412E and fails.

- If you specify CONTINUEIF, LOADPLUS issues message BMC51412E and terminates.

- If LOADPLUS invokes DSNUTILB when you specify FORMAT BMCUNLOAD, LOADPLUS issues message BMC50178E and terminates.

- LOADPLUS does not enforce check constraints when you specify FORMAT BMCUNLOAD. If ENFORCE CHECK CONSTRAINTS is in effect, LOADPLUS functions as if you specified ENFORCE NO. If ENFORCE CONSTRAINTS is in effect, LOADPLUS functions as if you specified ENFORCE RI.
  
  In either case, LOADPLUS issues message BMC50138I and continues processing. If check constraints are defined on the table and the value of the CHECKPEND option is YES, LOADPLUS sets CHKP following the load.

- If you specify any of the following LOAD command options with FORMAT BMCUNLOAD, LOADPLUS ignores the option:
  - ASCII/EBCDIC/UNICODE
  - CCSID
  - NOSUBS

Additional FORMAT BMCUNLOAD restrictions and considerations

This topic describes additional FORMAT BMCUNLOAD restrictions and considerations.

UNLOAD PLUS version

You cannot load input files generated by the FORMAT BMCLOAD option of UNLOAD PLUS by using a version of LOADPLUS that is earlier than the version of UNLOAD PLUS that you used. For example, if you unload your data by using FORMAT BMCLOAD in version 10.2 of UNLOAD PLUS, you cannot load it by using FORMAT BMCUNLOAD in version 10.1 of LOADPLUS.
LOB and XML data

LOADPLUS does not support loading LOB or XML data when you specify FORMAT BMCUNLOAD. LOADPLUS terminates in this case.

RULES installation option

For this type of load job, LOADPLUS changes the value of the RULES installation option to BMC, regardless of the value that was specified at installation.

ENUMROWS option

For this type of load job, BMC recommends that you do not specify ENUMROWS AUTO, but allow UNLOAD PLUS to supply a number for ENUMROWS. To enable this functionality, include a string for ENUMROWS when you specify the CNTLCARDS option in UNLOAD PLUS. When you do so, UNLOAD PLUS provides the number of unloaded rows to the ENUMROWS option in LOADPLUS. For an example, see “Example 14 LOAD REPLACE from UNLOAD PLUS data in internal format” on page 588.

Data translation

Data translation between the two tables is not supported. If you specify the CCSID option on your LOAD command, LOADPLUS ignores the option but loads the data.

Check constraints

For this type of load job, LOADPLUS does not validate check constraints, regardless of the value that you specify for the ENFORCE option. If there are check constraints on the table and the value of the CHECKPEND option is YES, LOADPLUS sets CHKP following the load.

**Note**

If you specify ENFORCE CONSTRAINTS, LOADPLUS does enforce referential constraints.

Identity columns

If any of the tables that are specified on your INTO statements contain an identity column and the tables are valid based on their verification records, LOADPLUS validates the identity column definition in the verification records of the input file against the identity column definition in the table that you are loading. If these definitions are not the same, LOADPLUS issues a warning message only if the value of the UPDATEMAXA or UPDMAXA option is YES. LOADPLUS loads the data regardless of whether the definitions are the same.
When you specify FORMAT BMCUNLOAD, LOADPLUS does not validate the identity column definition against the input data values. Therefore, you might have values in your identity column that are outside the range that you specified.

If LOADPLUS updates the value for MAXASSIGNEDVAL in SYSIBM.SYSSEQUENCES (when UPDATEMAXA YES is in effect), the value that LOADPLUS uses for MAXASSIGNEDVAL is the value for the source table that was in the DB2 catalog at the beginning of the unload process in UNLOAD PLUS. LOADPLUS updates MAXASSIGNEDVAL by using the same criteria as with any other load job.

**Discards**

To use a discard file from this type of load job as input to a subsequent load job, you must modify the file. For instructions, contact BMC Customer Support.

**Using control cards from UNLOAD PLUS after migrating data in a standard DB2 format**

When reloading data that you unloaded in a standard DB2 format by using UNLOAD PLUS, BMC recommends using the control cards that UNLOAD PLUS generated. However, you might need to make changes to those control cards under certain conditions.

For more information about the control cards that UNLOAD PLUS generates, see the *UNLOAD PLUS for DB2 Reference Manual*.

**Temporal tables**

When loading a system-period temporal table, you must make changes to the control cards that UNLOAD PLUS generated, depending on your requirements. The following table describes these changes:

**Table 29: Changes to UNLOAD PLUS generated control cards for temporal tables**

<table>
<thead>
<tr>
<th>If you want to load this type of data</th>
<th>Into this column</th>
<th>Make this change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input file data</td>
<td>System period begin</td>
<td>Add PERIODOVERRIDE to your INTO statement.</td>
</tr>
<tr>
<td></td>
<td>System period end</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transaction start ID</td>
<td>Add TRANSIDOVERRIDE to your INTO statement.</td>
</tr>
</tbody>
</table>
If you want to load this type of data

<table>
<thead>
<tr>
<th>Into this column</th>
<th>Make this change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generated defaults</td>
<td>System period begin System period end Transaction start ID</td>
</tr>
</tbody>
</table>

**Identity columns**

If the identity column in the table that you are loading is defined as GENERATED ALWAYS, you must add IDENTITYOVERRIDE YES.

**Considerations for BatchPipes input**

LOADPLUS allows you to run in an IBM BatchPipes environment, including the ability to use input from the Job Optimizer Pipes component of the BMC MainView Batch Optimizer product.

The following considerations apply when using data pipes:

- If your input data set is a pipe, LOADPLUS does not write discards to SYSDISC.

- You can specify a mixture of DASD files and pipes as input. However, you cannot concatenate these files within one SYSREC DD.

- If you use multiple pipe data sets, you must specify a separate SYSREC DD statement for each data set and the DCB parameters for all of the pipe data sets must be the same.

- If you are loading data with multiple pipes, each pipe must be generated by an independent job and each of these jobs must be able to run concurrently. For example, you cannot load data by using multiple pipes that are generated by multiple steps within a single job.

- You cannot specify a pipe data set with the INDSN command option.

- The automated file size estimation routine (AUTOENUMROWS=YES or ENUMROWS AUTO) cannot determine the number of rows from a pipe data set. If you are dynamically allocating work files or copy data sets and using pipe data sets as input, you must specify the appropriate numeric values with the ENUMROWS command option.
Considerations for table space compression

This topics in this section describe how LOADPLUS handles table space compression.

For LOAD REPLACE or LOAD RESUME YES PART n REPLACE, the value of the KEEPDICTIONARY installation option determines how LOADPLUS handles compression at a global level. However, you can override this value by specifying the KEEPDICTIONARY command option at either the table space or partition level. If you specify the KEEPDICTIONARY command option without a value, LOADPLUS assumes KEEPDICTIONARY YES, regardless of the value of the installation option. For more information, see “KEEPDICTIONARY” on page 188, “INTO TABLE” on page 306, and “KEEPDICTIONARY=NO” on page 735.

Considerations when using table space compression

The following additional considerations apply when you use table space compression with LOADPLUS:

- When both of the following conditions exist, specify a single subsystem ID (SSID) for your load job instead of a group attachment name. Ensure that the DB2 version of the SSID that you specify corresponds to the version of the DB2 load library that you specify in your STEPLIB.

  — You are running LOADPLUS in a data sharing environment.

  — The subsystems in your data sharing group are not all at the same DB2 version level.

- Loading a large number of compressed partitions might result in constrained resources. If you encounter this problem, consider specifying smaller groups of partitions in multiple load steps.

- When LOADPLUS invokes DSNUTILB, compression is handled by DSNUTILB. For information about how compression is handled for these jobs, see the documentation for the LOAD utility in the IBM DB2 Utility Guide and Reference.

- When you are loading a partition-by-growth table space and LOADPLUS adds partitions, the compression dictionary used for the new partitions depends on the type of load that you are running:

  — For LOAD REPLACE, LOADPLUS uses the dictionary from the first partition.

  — For LOAD RESUME YES, LOADPLUS uses the dictionary from the partition in which the load started adding rows.
LOADPLUS supports compression for XML data.

LOADPLUS reports the percentage of compressed rows in messages BMC50512I and BMC50522I. This percentage does not include rows that are compressed by an EDITPROC.

### Considerations when building a dictionary

LOADPLUS builds a new dictionary based on the following criteria:

- For LOAD REPLACE or LOAD RESUME YES PART n REPLACE jobs, LOADPLUS builds a new dictionary when any of the following conditions exists:
  - The value of KEEPDICTIONARY is NO.
  - The value of KEEPDICTIONARY is YES, and one of the following conditions is in effect:
    - A dictionary does not exist or the existing dictionary is unusable.
    - LOADPLUS is converting your data from BRF to RRF and the value of the DB2 subsystem parameter HONOR_KEEPDICTIONARY is NO.

- For LOAD RESUME YES jobs (without PART n REPLACE) on partitioned table spaces, LOADPLUS builds a new dictionary if the table space or partition is empty and a compression dictionary does not currently exist for the table space or partition. LOADPLUS builds a dictionary regardless of the value of the KEEPDICTIONARY option.

  **Note**

  For a partition-by-growth table space, if the first partition has a compression dictionary and any of the subsequent partitions are empty, LOADPLUS copies the dictionary from the first partition instead of building a new dictionary.

LOADPLUS builds the compression dictionary during the PRELOAD phase for a two-phase load and during the COMBINED phase for a single-phase load. After completely building the dictionary, LOADPLUS compresses the data. For a two-phase load, LOADPLUS stores the dictionary in the BMCDICT table between phases.

  **Note**

  LOADPLUS stores a dictionary in BMCDICT for each partition that you are loading. If you are loading a table space with a large number of partitions, you might need to increase the size of the BMCDICT table space from the standard size that was allocated during installation.

Note the following considerations when building a dictionary:

- LOADPLUS reserves enough pages for the largest dictionary size.
If LOADPLUS does not load enough rows to create a full dictionary, LOADPLUS
— Does not compress any rows
— Builds a non-optimal dictionary for future utility use

Considerations when keeping a dictionary

If a dictionary already exists, LOADPLUS keeps the existing dictionary and uses it
for compression when either of the following conditions exists:

■ For LOAD REPLACE or LOAD RESUME YES PART \( n \) REPLACE jobs, if the value
  of KEEPDICTIONARY is YES

■ For LOAD RESUME YES jobs (without PART \( n \) REPLACE)

  **Note**
  If the existing dictionary is unusable, LOADPLUS continues but does not
  compress any rows.

Indexes on expression

LOADPLUS natively supports reorganizing indexes that contain a key derived from
certain expressions, and reorganizing table spaces that contain those indexes.

With the exceptions described in “Restrictions on indexes on expression” on page
143, the utility currently supports expressions that contain the following SQL
functions:

■ UPPER
■ LOWER
■ SUBSTR
■ DATE
■ DAY
■ MONTH
■ YEAR

For all other functions that IBM supports for indexes on expression, the utility
invokes DSNUTILB.
Restrictions on indexes on expression

When any of the following conditions exists, the utility does not provide native support and invokes DSNUTILB instead:

■ A SUBSTR function has any of the following characteristics:
  — The function is on a numeric column.
  — The function uses columns in the table space to supply start and end values.
  — The function is on a LOB column (for example, an index on an inline LOB column).

■ An expression contains nested columns.

■ An UPPER or LOWER function has any of the following characteristics:
  — The function includes a length parameter.
  — The locale name has a value other than UNI or blank.
  — The column contains mixed or DBCS data, and the locale name is blank.
  — For ASCII and EBCDIC objects, the column contains mixed data.

■ A DATE, DAY, MONTH, or YEAR function has any of the following characteristics:
  — The function references more than one column.
  — The column data type is not DATE, TIMESTAMP, or TIMESTAMP WITH TIME ZONE.

Considerations for reordered row format

This topic describes considerations that apply to reordered row format (RRF) in LOADPLUS.
### Table 30: Considerations for reordered row format

<table>
<thead>
<tr>
<th>Option</th>
<th>Additional condition</th>
<th>Consideration</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD REPLACE or LOAD RESUME YES PART n REPLACE (considerations apply to the partitions that are being replaced)</td>
<td>VALIDPROC or EDITPROC on the table</td>
<td>LOADPLUS uses the existing row format of the table space.</td>
</tr>
<tr>
<td></td>
<td>Partition-by-growth table space</td>
<td>LOADPLUS uses the existing row format of the table space. The following requirements apply:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ All partitions in the table space must be in the same row format.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ The partitions must be in the following format, based on the DB2 subsystem parameter RRF:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— When RRF=ENABLE, the format must be RRF.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— When RRF=DISABLE, the format must be BRF.</td>
</tr>
<tr>
<td></td>
<td>FORMAT BMCUNLOAD</td>
<td>LOADPLUS uses the existing row format of the table space. The following requirements apply:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ The row format on the target table space must be the same as the row format on the source.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ All partitions in the table space must be in the same row format.</td>
</tr>
<tr>
<td></td>
<td>All other conditions</td>
<td>LOADPLUS converts data from BRF to RRF. The following considerations apply:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ LOADPLUS does not convert objects from BRF to RRF when running on a subsystem that has been migrated directly from DB2 Version 8 to DB2 Version 10, but has not yet been migrated to DB2 Version 10 new-function mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ If a table space is compressed, LOADPLUS builds a new dictionary except when both of the following options are in effect:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— The value of the LOADPLUS KEEPDICTIONARY option is YES.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— The value of the DB2 subsystem parameter HONOR_KEEPDICTIONARY is YES.</td>
</tr>
<tr>
<td>Option</td>
<td>Additional condition</td>
<td>Consideration</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>LOAD RESUME YES</td>
<td>Partition-by-growth table</td>
<td>LOADPLUS uses the existing row format of the table space. The following</td>
</tr>
<tr>
<td></td>
<td>space</td>
<td>requirements apply:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ All partitions in the table space must be in the same row format.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ The partitions must be in the following format, based on the DB2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>subsystem parameter RRF:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— When RRF=ENABLE, the format must be RRF.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— When RRF=DISABLE, the format must be BRF.</td>
</tr>
<tr>
<td>FORMAT BMCUNLOAD</td>
<td></td>
<td>LOADPLUS uses the existing row format of the table space. The following</td>
</tr>
<tr>
<td></td>
<td></td>
<td>requirements apply:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ The row format on the target table space must be the same as the row format</td>
</tr>
<tr>
<td></td>
<td></td>
<td>on the source.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ All partitions in the table space must be in the same row format.</td>
</tr>
<tr>
<td>All other conditions</td>
<td></td>
<td>LOADPLUS uses the existing row format of the table space.</td>
</tr>
</tbody>
</table>

**Image copies and recoverability of the loaded table space**

If you do not take an image copy as part of the LOADPLUS job, you must take some action to ensure that DB2 can recover the table space, if necessary, after the load job.

If you want an image copy, run COPY PLUS or the IBM DB2 COPY utility. You can also run DSN1COPY or any other suitable substitute that your system provides.

**Note**

When performing an SQLAPPLY load, the table space is always recoverable, assuming that it was recoverable before the load job.
LOADPLUS optionally enforces DB2-defined referential constraints, check constraints, and informational referential constraints.

**Referential constraints and informational referential constraints**

LOADPLUS enforces referential constraints differently, depending on the type of load job that you are running.

In all cases, referential constraints are checked between parent and child tables only when you are loading the child table.

**Referential constraints and SQLAPPLY load jobs**

The High-speed Apply architecture always preserves referential integrity relationships during an SQLAPPLY load job. Therefore, when you specify the ENFORCE option with an SQLAPPLY load job, LOADPLUS ignores that option.

*Note*
SQLAPPLY load jobs do not enforce informational referential constraints.

When referential integrity is involved, BMC recommends that you specify ORDER YES, causing LOADPLUS to sort your data by table. This sort order causes LOADPLUS to load parent tables before their child tables.

If you attempt to load tables that are self-referencing or that have circular relationships and you specify ORDER YES, LOADPLUS cannot determine the correct order for an SQLAPPLY load. In this case, LOADPLUS issues warning message BMC51582W but continues processing. When processing passes to DB2, you might receive SQL-530 errors. To maintain referential integrity for this type of load, order the data before your load job, and specify ORDER NO with your LOAD command.

**Referential constraints and DSNUTILB load jobs**

When LOADPLUS invokes DSNUTILB, referential integrity relationships are handled by DSNUTILB based on the ENFORCE option that you specify.

However, the IBM DB2 LOAD utility ENFORCE options are CONSTRAINTS and NO. If you specify the LOADPLUS option ENFORCE RI or ENFORCE CONSTRAINTS INFORI YES, LOADPLUS cannot convert that option to an
appropriate IBM option. In these cases, LOADPLUS issues message BMC50178E and terminates.

**Enforcing referential constraints for other load types**

For other load types, LOADPLUS enforces referential constraints based on the ENFORCE option that is in effect.

When enforcing referential constraints, LOADPLUS verifies the constraints and discards any violations.

**To enforce referential constraints**

1. Ensure that a currently supported version of XBM or SUF (as described in “LOADPLUS system and software requirements” on page 49) is installed and available to LOADPLUS.

   XBM and SUF require a started task.

2. Enable XBM objects:

   a. Create and activate the XBM management set that contains the appropriate snapshot object definition.

   b. Create and activate the XBM configuration that contains the appropriate cache attributes.

   For proper LOADPLUS operation, ensure that XBM or SUF is configured with sufficient cache for the number of concurrent snapshots and the level of DB2 update activity at your site. For specific cache configuration information, see the *EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide*.

3. Specify the following options in your LOADPLUS installation options module or on your LOAD command:

   - (optional) XBMID, if you want to use a specific XBM subsystem for snapshot processing
   - PRELOAD LOAD
   - Either ENFORCE RI (to check only referential constraints) or ENFORCE CONSTRAINTS (to check both referential and check constraints)

   You can optionally specify INFORI YES on either of these options to tell LOADPLUS to also enforce informational referential constraints. For more information about these options, see “ENFORCE” on page 263.
4 Ensure that you have READ access on the primary index of the parent table for the table that you are loading.

5 Ensure that your objects and the options on your LOAD command meet the requirements described in “Restrictions on enforcing referential constraints” on page 148.

Restrictions on enforcing referential constraints

When checking referential constraints during your load job, LOADPLUS has the following restrictions on your objects. When encountering any of these conditions, LOADPLUS issues an appropriate message and terminates.

- You cannot load the parent table and child table in the same load execution.
- You cannot load multiple tables concurrently if those tables use the same parent index.
- Primary indexes on the parent table of the table that you are loading cannot be in any restrictive status (for example, RBDP).
- Primary indexes on the parent table of the table that you are loading must already be defined. (In other words, if the index space was defined with DEFINE NO, it must be materialized before the load job.)
- Objects cannot be in LPL or WEPR status.
- Identity columns in the tables that you are loading cannot have check constraints defined on them.

Considerations for check constraints

For most load jobs, LOADPLUS enforces DB2-defined check constraints based on the ENFORCE option that is in effect.

When enforcing constraints, LOADPLUS verifies the constraints and discards any violations. For more information about the ENFORCE options, see “ENFORCE” on page 263.

SQLAPPLY load jobs

For SQLAPPLY load jobs, if DB2-defined check constraints exist for a table that you are loading, the High-speed Apply architecture automatically checks these constraints and discards any violations. LOADPLUS ignores the ENFORCE option for this type of load job.
DSNUTILB load jobs

When LOADPLUS invokes DSNUTILB, check constraint checking is handled by DSNUTILB. However, the IBM DB2 LOAD utility ENFORCE options are CONSTRAINTS and NO. If you specify ENFORCE CHECK CONSTRAINTS, LOADPLUS cannot convert that option to an appropriate IBM option, so LOADPLUS issues message BMC50178E and terminates.

FORMAT BMCUNLOAD

For all types of load jobs except an SQLAPPLY load, LOADPLUS does not verify check constraints when you specify FORMAT BMCUNLOAD.

Note

If you specify ENFORCE CONSTRAINTS in this case, LOADPLUS checks only referential constraints.

DEFINE NO objects

If you are loading a table whose table space or index spaces are defined with DEFINE NO and a check constraint exists on the table, LOADPLUS attempts to build a row that satisfies the check constraint while materializing the table. However, if LOADPLUS cannot resolve the check constraint, LOADPLUS issues messages BMC51492W, which describes the check constraint that it cannot resolve, and BMC51491E, then terminates.

To load this table, you must manually materialize the associated data sets and restart the load job.

Identity columns

If you are loading an identity column that either contains a check constraint or is referenced in the check condition of a DB2 check constraint and both of the following conditions also exist, LOADPLUS terminates:

- ENFORCE CONSTRAINTS or ENFORCE CHECK CONSTRAINTS is in effect.
- The identity column definition and LOADPLUS field specification for the identity column could result in LOADPLUS generating values.

Constraint content restrictions

The following restrictions apply to the content of your check constraints:
When you use a signed token, do not place a space between the sign and the value.

If check constraint conditions contain one or more of the following constant types, LOADPLUS terminates:

- Floating-point constants
- Constants whose length is greater than 254 bytes

**Considerations for CHKP status**

When LOADPLUS does not enforce all referential and check constraints on the table that you are loading, the CHEKPEND installation option or CHECKPEND command option tell LOADPLUS whether to set CHKP status on that table space.

**DSNUTILB load**

When LOADPLUS invokes DSNUTILB, setting pending statuses is handled by DSNUTILB. For information about how CHKP status is handled for these jobs, see the documentation for the LOAD utility in the IBM DB2 Utility Guide and Reference.

**LOAD REPLACE**

Note the following additional considerations about how LOADPLUS sets and resets CHKP status when you specify LOAD REPLACE. LOADPLUS sets CHKP status only when the CHECKPEND or CHEKPEND option is YES.

- If LOADPLUS successfully checks all constraints on the table that you are loading, LOADPLUS resets CHKP status on that table space, if it was in CHKP status before the load job.

- If LOADPLUS successfully checks all constraints on the table that you are loading, LOADPLUS sets CHKP status on any child table spaces of that table unless both of the following conditions exist:
  - The child table is in the same table space as the table that you are loading.
  - The load job emptied the child table.

  **Note**

  If both of these conditions exist, LOADPLUS resets CHKP on the child table space, if it was in CHKP status before the load job.

- If the load job empties a table that is in the same table space as the table that you are loading, LOADPLUS sets CHKP status on the child table spaces of the table that was emptied.
If the load job empties one or more partitions in the table space that you are loading, LOADPLUS resets CHKP status for those partitions.

**LOAD RESUME YES**

When you specify an ENFORCE option to enforce constraints, LOADPLUS checks only the rows that are added by the load job. Because existing rows are not checked, the table space remains in CHKP status if it was in CHKP status before the load job.

**TABLE REPLACE**

Note the following additional considerations about how LOADPLUS sets and resets CHKP status when you specify LOAD RESUME YES TABLE tableName REPLACE:

- LOADPLUS sets CHKP status on any child table spaces of that table.
- If you do not load all tables in the table space, that table space remains in CHKP status if it was in CHKP status before the load job.

**Restoring integrity**

If LOADPLUS sets CHKP status, you must take additional steps to ensure that referential integrity or table check integrity has not been violated. After the LOADPLUS utility completes, you must run CHECK PLUS (specifying the CHECK DATA command) or the IBM CHECK DATA utility.

However, if LOADPLUS sets CHKP status on a table because it enforced informational referential constraints on the parent table, you must use one of the following options to restore integrity:

- Run LOADPLUS with LOAD REPLACE on the table in CHKP status.
- If the table in CHKP status has no constraints other than informational constraints, run CHECK PLUS (specifying the CHECK DATA command) on the table.

To prevent this condition from occurring when you are enforcing only informational constraints, BMC recommends that you specify CHECKPEND NO.

LOADPLUS allows an image copy to be made after a table is loaded, even when the table is in CHKP status. Because both check utilities log any changes that are made to the table, a full RECOVER utility execution that uses the copy applies the log records, thus ensuring the integrity of the data in the table. A RECOVER TOCOPY utility execution places the affected table in CHKP status again. If the table is in CHKP status due to referential integrity constraints, a subsequent invocation of either check utility restores referential integrity.
DB2 user exits with LOADPLUS

LOADPLUS can invoke the following types of DB2 user exits during processing:

- EDITPROCs
- VALIDPROCs
- FIELDPROCs
- Date exits:
  - DSNXVDTX (EBCDIC date exit routine)
  - DSNXVDTA (ASCII date exit routine)
  - DSNXVDTU (Unicode date exit routine)
- Time exits:
  - DSNXVTMX (EBCDIC time exit routine)
  - DSNXVTMA (ASCII time exit routine)
  - DSNXVTMU (Unicode time exit routine)
- Authorization exit

By default, LOADPLUS invokes these exits in supervisor state (and PSW key=7).

DB2 real-time statistics in LOADPLUS

LOADPLUS automatically updates the DB2 real-time statistics tables during the UTILTERM phase.

The columns that LOADPLUS updates depends on the type of load job that you are running. The following table describes where to find the information about the columns that LOADPLUS updates and the value with which it updates those columns.

Note

When LOADPLUS invokes DSNUTILB, updating real-time statistics tables is handled by DSNUTILB. Therefore, the information in this section about how the tables are updated does not apply to this type of load job.
Table 31: How LOADPLUS updates real-time statistics tables

<table>
<thead>
<tr>
<th>Load type</th>
<th>Description or reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD RESUME NO or LOAD REPLACE</td>
<td>LOADPLUS updates the columns as described in Table 32 on page 154.</td>
</tr>
<tr>
<td>LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY or LOAD RESUME YES SHRLEVEL CHANGE without PART n REPLACE</td>
<td>Because SQLAPPLY load jobs use SQL insert processing, DB2 updates the real-time statistics tables as it would for any other DB2 application that uses SQL insert processing.</td>
</tr>
<tr>
<td>LOAD RESUME YES INTO TABLE tableName REPLACE</td>
<td>LOADPLUS does not currently update the real-time statistics tables for this type of load job.</td>
</tr>
<tr>
<td>LOAD RESUME YES INDEX BUILD PART n REPLACE</td>
<td>LOADPLUS updates the columns as described in Table 32 on page 154 for each partition that you are replacing.</td>
</tr>
</tbody>
</table>
| LOAD RESUME YES INDEX UPDATE PART n REPLACE                               | • LOADPLUS updates the columns as described in Table 32 on page 154 for the table space and index space partitions that you are replacing.  
                                        • LOADPLUS updates the columns as described in the INDEX UPDATE information in Table 33 on page 156 for the nonpartitioned indexes. |
| LOAD RESUME YES (all others)                                              | LOADPLUS updates the columns as described in Table 33 on page 156.                       |

The following information applies when updating the real-time statistics tables:

- If LOADPLUS cannot update a column, processing continues and LOADPLUS issues message BMC50290I.

- When LOADPLUS stops the object, DB2 updates real-time statistics values with the most current accumulated data. LOADPLUS updates those values again during the UTILTERM phase to reflect the effect of the load operation.

- LOADPLUS jobs can affect the accuracy of the statistics in the same way as IBM DB2 LOAD utility jobs can. For more information about the accuracy of these statistics, see the IBM documentation.

LOADPLUS does not update real-time statistics for any of the following conditions:

- When you specify LOAD RESUME YES INTO TABLE tableName REPLACE
- For table space and index partitions that do not participate in the load
- For indexes that do not participate in the load (because you specified SKIPIX)

The following tables describe the DB2 real-time statistics table columns that LOADPLUS updates and the values with which it updates those columns.
LOADPLUS does not update any columns that are not listed in these tables.

Table 32: Real-time statistics updates for LOAD RESUME NO, LOAD REPLACE, and LOAD RESUME YES PART n REPLACE

<table>
<thead>
<tr>
<th>Table</th>
<th>Column</th>
<th>Updated value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSIBM.SYSTABLESPACESTATS</td>
<td>UPDATESTATSTIME</td>
<td>Timestamp of the update</td>
</tr>
<tr>
<td></td>
<td>TOTALROWS</td>
<td>Number of rows loaded</td>
</tr>
<tr>
<td></td>
<td>NACTIVE</td>
<td>Number of pages loaded</td>
</tr>
<tr>
<td></td>
<td>NPAGES</td>
<td>Number of pages loaded that contain rows</td>
</tr>
<tr>
<td></td>
<td>SPACE</td>
<td>Value from the VSAM catalog</td>
</tr>
<tr>
<td></td>
<td>EXTENTS</td>
<td>Value from the VSAM catalog</td>
</tr>
<tr>
<td></td>
<td>LOADRLASTTIME</td>
<td>Timestamp of the update</td>
</tr>
<tr>
<td></td>
<td>REORGINSERTS a</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGDELETES a</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGUPDATES a</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGDISORGLOB a</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGUNCLUSTINS a</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGMASSDELETE a</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGNEARINDREF a</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGFARINDREF a</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>COPYLASTTIME b</td>
<td>Timestamp of the update</td>
</tr>
<tr>
<td></td>
<td>COPYUPDATEDPAGES b,c</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>COPYCHANGES b,c</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>COPYUPDATERLSN b,c</td>
<td>Null</td>
</tr>
<tr>
<td></td>
<td>COPYUPDATETIME b,c</td>
<td>Null</td>
</tr>
<tr>
<td></td>
<td>DATASIZE</td>
<td>Total data loaded, in bytes</td>
</tr>
</tbody>
</table>
Table: DB2 real-time statistics in LOADPLUS

<table>
<thead>
<tr>
<th>Table</th>
<th>Column</th>
<th>Updated value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSIBM.SYSINDEXSPACESTATS</td>
<td>UPDATESTATSTIME</td>
<td>Timestamp of the update</td>
</tr>
<tr>
<td>Note: LOADPLUS updates the</td>
<td>TOTALENTRIES</td>
<td>Number of rows loaded</td>
</tr>
<tr>
<td>columns for nonpartitioned</td>
<td></td>
<td></td>
</tr>
<tr>
<td>indexes of a LOAD RESUME YES</td>
<td>NLEVELS</td>
<td>Number of index levels</td>
</tr>
<tr>
<td>INDEX UPDATE PART n REPLACE</td>
<td>NLEAF</td>
<td>Number of leaf pages</td>
</tr>
<tr>
<td>load with different</td>
<td>NACTIVE</td>
<td>Number of pages loaded</td>
</tr>
<tr>
<td>information. You can find</td>
<td>SPACE</td>
<td>Value from the VSAM catalog</td>
</tr>
<tr>
<td>this information in the</td>
<td>EXTENTS</td>
<td>Value from the VSAM catalog</td>
</tr>
<tr>
<td>INDEX UPDATE column of</td>
<td>LOADRLASTTIME</td>
<td>Timestamp of the update</td>
</tr>
<tr>
<td>Table 33 on page 156.</td>
<td>REORGINserts</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGDELETES</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGPAPPENDINSERT</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGPSEUDELETES</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGmassDELETE</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGlefAFNEAR</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGlefAFFAR</td>
<td>Zero</td>
</tr>
<tr>
<td></td>
<td>REORGNUMLEVELS</td>
<td>Zero</td>
</tr>
</tbody>
</table>

a If REORGLASTTIME in SYSIBM.SYSTABLESPACESTATS is null, LOADPLUS does not update these columns.

b LOADPLUS updates these columns only if the load job registers a copy.

c If COPYLASTTIME in SYSIBM.SYSTABLESPACESTATS is null or if you specify COPY NO, LOADPLUS does not update these columns.

d If REORGLASTTIME in SYSIBM.SYSINDEXSPACESTATS is null, LOADPLUS does not update these columns.

The following table does not apply to some types of LOAD RESUME YES jobs. For more information, see Table 31 on page 153.
Table 33: Real-time statistics updates for LOAD RESUME YES

<table>
<thead>
<tr>
<th>Table</th>
<th>Column</th>
<th>Updated value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSIBM.SYSTABLESPACESTATS</td>
<td>UPDATESTATSTIME</td>
<td>Timestamp of the update</td>
</tr>
<tr>
<td></td>
<td>TOTALROWS</td>
<td>Number of rows loaded plus the previous value</td>
</tr>
<tr>
<td></td>
<td>NACTIVE&lt;sup&gt;a&lt;/sup&gt;</td>
<td>Number of pages loaded plus the previous value</td>
</tr>
<tr>
<td></td>
<td>NPAGES&lt;sup&gt;a&lt;/sup&gt;</td>
<td>Number of pages loaded that contain rows plus the previous value</td>
</tr>
<tr>
<td></td>
<td>SPACE&lt;sup&gt;a&lt;/sup&gt;</td>
<td>Value from VSAM catalog</td>
</tr>
<tr>
<td></td>
<td>EXTENTS&lt;sup&gt;a&lt;/sup&gt;</td>
<td>Value from VSAM catalog</td>
</tr>
<tr>
<td></td>
<td>COPYLASTTIME&lt;sup&gt;b&lt;/sup&gt;</td>
<td>Timestamp of the update</td>
</tr>
</tbody>
</table>
|                        | COPYUPDATEDPAGES<sup>c</sup> | ■ If a copy is registered: zero  
■ If a copy is not registered: number of pages loaded plus the previous value |
|                        | COPYCHANGES<sup>c</sup>  | ■ If a copy is registered: zero  
■ If a copy is not registered: number of rows loaded plus the previous value |
<p>|                        | COPYUPDATELRSN&lt;sup&gt;c&lt;/sup&gt; | Null                                                                          |
|                        | COPYUPDATETIME&lt;sup&gt;c&lt;/sup&gt; | Null                                                                         |
|                        | DATASIZE         | Total data loaded, in bytes, plus the previous value                         |</p>
<table>
<thead>
<tr>
<th>Table</th>
<th>Column</th>
<th>Updated value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSIBM.SYSINDEXSPACESTATS</td>
<td>UPDATESTATSTIME</td>
<td>Timestamp of the update</td>
</tr>
<tr>
<td></td>
<td>TOTALENTRIES</td>
<td>Number of rows loaded plus the previous value</td>
</tr>
<tr>
<td></td>
<td>NLEVELS</td>
<td>■ INDEX BUILD: number of index levels</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ INDEX UPDATE: not updated</td>
</tr>
<tr>
<td></td>
<td>NLEAF</td>
<td>■ INDEX BUILD: number of leaf pages</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ INDEX UPDATE: not updated</td>
</tr>
<tr>
<td></td>
<td>NACTIVE</td>
<td>■ INDEX BUILD: number of pages loaded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ INDEX UPDATE: not updated</td>
</tr>
<tr>
<td></td>
<td>SPACE</td>
<td>■ INDEX BUILD: value from VSAM catalog</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ INDEX UPDATE: not updated</td>
</tr>
<tr>
<td></td>
<td>EXTENTS</td>
<td>■ INDEX BUILD: value from VSAM catalog</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ INDEX UPDATE: not updated</td>
</tr>
</tbody>
</table>

1. LOADPLUS does not update this column if TOTALROWS is null.
2. LOADPLUS updates this column only if the load job registers a copy.
3. If COPYLASTTIME is null, LOADPLUS does not update these columns.
4. LOADPLUS does not update any columns in this table if TOTALENTRIES is null and you specified INDEX UPDATE.
Return code hierarchy in LOADPLUS

LOADPLUS assigns return codes to certain conditions that occur during the execution of the load job, such as successful completion with no warnings or errors (return code 0) or setting COPY pending (return code 4).

For return code explanations, access the messages section of the BMC Documentation Center; you can access the Documentation Center from the BMC Support Central site (http://www.bmc.com/support).

In addition, by using installation or command options, you can define specific return codes that you want LOADPLUS to assign for certain conditions. The following table describes these options and where to find detailed information about them.

Table 34: User-defined return codes

<table>
<thead>
<tr>
<th>Condition</th>
<th>Command option and reference</th>
<th>Installation option and reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job completes successfully but one or more records are discarded</td>
<td>IFDISCARDS “IFDISCARDS” on page 223</td>
<td>DISCARDRC “DISCARDRC=0” on page 722</td>
</tr>
<tr>
<td>PRELOAD PAUSE job completes successfully but one or more records are discarded</td>
<td>PRELOAD PAUSE ANYDISCARDS RETCODE “PRELOAD” on page 208</td>
<td>PAUSEDISCARDRC “PAUSEDISCARDRC=4” on page 742</td>
</tr>
<tr>
<td>Number of discard violations reaches the limit specified on DISCARDS option</td>
<td>IFDISCARDLIMIT “IFDISCARDLIMIT” on page 224</td>
<td>DISCARDLIMRC “DISCARDLIMRC=8” on page 722</td>
</tr>
<tr>
<td>No records are loaded</td>
<td>IFZEROROWS “IFZEROROWS” on page 225</td>
<td>ZEROROWRC “ZEROROWRC=0” on page 759</td>
</tr>
</tbody>
</table>

When the job completes, LOADPLUS ends with the highest return code that it assigned during that job and reports this return code in the SYSPRINT output. Due to other conditions detected during the load job, the final return code might not be the value that you specified for one of the user-defined return code options. The following table illustrates final return codes for potential combinations of conditions:

Table 35: Return code hierarchy examples

<table>
<thead>
<tr>
<th>Options specified</th>
<th>Additional conditions</th>
<th>Job completion status</th>
<th>Winning condition</th>
<th>Final return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFDISCARDS 3</td>
<td>One or more records are discarded.</td>
<td>The job completes successfully.</td>
<td>COPY NO COPYPEND YES</td>
<td>4</td>
</tr>
<tr>
<td>COPY NO COPYPEND YES</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Options specified</td>
<td>Additional conditions</td>
<td>Job completion status</td>
<td>Winning condition</td>
<td>Final return code</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
<td>-------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>IFDISCARDS 3 ENFORCE NO CHECKPEND YES</td>
<td>■ Check constraints are defined on the table, and the job is not an SQLAPPLY load.</td>
<td>The job completes successfully.</td>
<td>CHECKPEND YES</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>■ Records are discarded.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IFDISCARDS 3 SKIPIX SIX</td>
<td>One or more records are discarded.</td>
<td>The job completes successfully.</td>
<td>SKIPIX SIX, because it sets nonpartitioned indexes to REBUILD pending status</td>
<td>4</td>
</tr>
<tr>
<td>IFDISCARDS 3 IFZEROROWS 7</td>
<td>All records are discarded.</td>
<td>The job completes successfully.</td>
<td>IFZEROROWS</td>
<td>7</td>
</tr>
<tr>
<td>IFDISCARDS 7 IFZEROROWS 3</td>
<td>All records are discarded.</td>
<td>The job completes successfully.</td>
<td>IFDISCARDS</td>
<td>7</td>
</tr>
<tr>
<td>IFDISCARDS 7 PRELOAD PAUSE ANYDISCARDS RETCODE 5</td>
<td>A record is discarded.</td>
<td>The job pauses at the end of the PRELOAD phase, a</td>
<td>RETCODE</td>
<td>5</td>
</tr>
<tr>
<td>IFZEROROWS 3 FORMAT BMCUNLOAD</td>
<td>The input file contains only invalid verification records (normally return code 0), so no rows are loaded.</td>
<td>The job completes successfully.</td>
<td>IFZEROROWS</td>
<td>3</td>
</tr>
<tr>
<td>IFDISCARDLIMIT 10 DISCARDS 1 PRELOAD PAUSE ANYDISCARDS RETCODE 20</td>
<td>A record is discarded.</td>
<td>The job ends when the discard limit is reached. b</td>
<td>IFDISCARDLIMIT</td>
<td>10</td>
</tr>
</tbody>
</table>

a IFDISCARDS is not in effect for a paused job. If records are discarded after you restart the job, the job ends with return code 7 (the return code specified with IFDISCARDS in this example).

b ANYDISCARDS RETCODE 20 applies only if the job completes the PRELOAD phase and the job pauses. In this case, LOADPLUS reached the discard limit before completing the PRELOAD phase, causing the return code for IFDISCARDLIMIT to take effect.
Image copies after recovery in LOADPLUS

The LOADPLUS copy function, when used with the LOAD RESUME YES option, provides performance benefits by not resetting the modified-page indicators in the table space data pages and space maps on existing pages, thus substantially reducing I/O operations.

Other products’ options (such as the RESETMOD NO option in COPY PLUS) also use this technique and do not reset the modified-page indicators in the table space.

Because of the way DB2 handles these modified-page indicators, you need to routinely make a full image copy of any table space that you recover by using this type of image copy before continuing to make incremental image copies. You should make the full image copy by using COPY PLUS with the RESETMOD YES option specified or the DB2 COPY utility.

XBM and SUF considerations in LOADPLUS

To enhance performance, LOADPLUS uses several features of XBM or SUF during portions of the load process.

If you have installed the required version of XBM or SUF, LOADPLUS can use the following XBM or SUF functions:

- Instant Snapshot functionality for a LOAD RESUME YES SHRLEVEL REFERENCE job
- Software snapshot functionality for referential constraint checking
- zIIP processing

XBM and SUF are licensed, installed, and maintained separately from LOADPLUS. 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 XBM features.
- A license for SUF authorizes you to use only the snapshot and zIIP-processing features of XBM.

**Note**

If you are licensed only for a BMC solution that contains LOADPLUS, your license authorizes you to use SUF, not the full version of XBM.
For more information, see the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide.

Considerations for using XBM or SUF with LOAD RESUME YES SHRLEVEL REFERENCE

When performing a LOAD RESUME YES SHRLEVEL REFERENCE load, LOADPLUS uses the Instant Snapshot functionality of XBM or SUF to create a consistent copy of the participating table space or partitions. LOADPLUS uses this copy as the staging data set for the load process.

For more information, including the procedure and requirements for running this type of load job, see “Running SHRLEVEL REFERENCE” on page 89.

Considerations for using XBM or SUF to enforce referential constraints

LOADPLUS uses the software snapshot functionality of XBM or SUF to make a consistent image of the primary indexes on the parent table of the table that you are loading, without making those indexes unavailable.

If you do not have an appropriate version of XBM or SUF installed and available to LOADPLUS, specifying ENFORCE RI or ENFORCE CONSTRAINTS causes LOADPLUS to terminate.

For more information, including the procedure for enforcing referential constraints, see “Referential and check constraints” on page 146.

Using XBM or SUF to enable zIIP processing

You can use a component of XBM or SUF to enable zIIP processing during your load job.

To enable zIIP processing

1. Ensure that you have a version of XBM or SUF that supports zIIP processing, as described in “LOADPLUS system and software requirements” on page 49.
2. Ensure that an XBM subsystem is started.
3 (optional) Use the XBMID installation or command option to specify the XBM subsystem that you want LOADPLUS to access.

4 Ensure that the value of the ZIIP installation or command option is YES.

## DB2 features that LOADPLUS does not support

This version of LOADPLUS has the following limitations on support for the features of DB2:

- This version of LOADPLUS does not support the following objects or features:
  - IBM FlashCopy image copies
  - Row- and column-level security
  - Loading data in IEEE Binary Floating Point (BFP) format
  - Spatial data
  - Unicode object names
  - Use of striped data sets for DB2 VSAM objects
  - Using TEMPLATE control statements
    However, LOADPLUS provides the DSNPAT installation or command option, which allows you to specify a data set name pattern for your dynamically allocated data sets.

- This version of LOADPLUS does not update the modified-page indicators in the space map pages that DB2 tracks when an object is defined with TRACKMOD YES.

- This version of LOADPLUS supports extended RBA and LRSN values, but does not convert values from basic format to extended format.

- As with the IBM DB2 LOAD utility, running LOADPLUS on encrypted data might produce unpredictable results.
Syntax of the LOAD command

This chapter describes the command options that you can use with the LOAD command.

Command syntax rules for LOADPLUS

The following general rules apply to the command syntax for the LOADPLUS for DB2 product:

- In a LOAD command, LOADPLUS considers any line beginning with an asterisk (*) in column 1 to be a comment and ignores it.

- When encountering two consecutive hyphens in a line (except within a delimited token), LOADPLUS considers the remainder of the line to be a comment, which LOADPLUS ignores.

- When you use a signed token, do not place a space between the sign and the value.

- You can split a token (such as a keyword, identifier, or constant) across a line. However, LOADPLUS ignores anything in columns 73 through 80.

A blank, or a delimiter if the token is delimited, indicates the end of the token. If an undelimited token ends in column 72, column 1 on the next line must be blank.

**Note**

If you use applications that automate JCL submission and resolve symbolic variables within your JCL, the resulting control cards might not appear as they do in the JCL that you created. These applications might produce an invalid command statement.

- If you specify the same command option more than once, LOADPLUS uses only the last option that you specify. For example, if you specify the following options, LOADPLUS accepts ORDER YES as the processing option:

```
LOAD DATA
ORDER NO ORDER YES
```
Specifying object names in your LOADPLUS syntax

In addition to the token information in the syntax rules section, the following information applies to object names that you specify in your LOADPLUS syntax:

- LOADPLUS does not support Unicode table names.
- LOADPLUS does not support delimited object names that do not have a character representation in EBCDIC.

Alphabetical listing of LOADPLUS command options

The table in this topic lists the LOADPLUS command options in alphabetical order.

This table provides a reference to description and usage information about each option.

Table 36: LOADPLUS command options

<table>
<thead>
<tr>
<th>Command options</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVE</td>
<td>“ACTIVE” on page 398</td>
</tr>
<tr>
<td>ANALYZE</td>
<td>“ANALYZE” on page 258</td>
</tr>
<tr>
<td>APCOLLECTION</td>
<td>“CHANGE” on page 190</td>
</tr>
<tr>
<td>APCommit</td>
<td>“CHANGE” on page 190</td>
</tr>
<tr>
<td>APMAXAGENTS</td>
<td>“CHANGE” on page 190</td>
</tr>
<tr>
<td>APMULTIROW</td>
<td>“CHANGE” on page 190</td>
</tr>
<tr>
<td>APOWNER</td>
<td>“CHANGE” on page 190</td>
</tr>
<tr>
<td>APRETRYLIM</td>
<td>“CHANGE” on page 190</td>
</tr>
<tr>
<td>APRETRYVAL</td>
<td>“CHANGE” on page 190</td>
</tr>
<tr>
<td>ASCII</td>
<td>“ASCII” on page 204</td>
</tr>
<tr>
<td>AVAILPAGEPCT</td>
<td>“AVAILPAGEPCT” on page 237</td>
</tr>
<tr>
<td>Command options</td>
<td>Reference</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------</td>
</tr>
<tr>
<td>AVGVOLSP</td>
<td>“AVGVOLSP” on page 409</td>
</tr>
<tr>
<td>BMCSTATS</td>
<td>“BMCSTATS” on page 289</td>
</tr>
<tr>
<td>CCSID</td>
<td>“CCSID” on page 204</td>
</tr>
<tr>
<td>CENTURY</td>
<td>“CENTURY” on page 272</td>
</tr>
<tr>
<td>CHECKPEND</td>
<td>“CHECKPEND” on page 267</td>
</tr>
<tr>
<td>CONTINUEIF</td>
<td>“CONTINUEIF” on page 243</td>
</tr>
<tr>
<td>COPY</td>
<td>“COPY” on page 294</td>
</tr>
<tr>
<td>COPYDDN</td>
<td>“COPYDDN” on page 300</td>
</tr>
<tr>
<td>COPYLVL</td>
<td>“COPYLVL” on page 299</td>
</tr>
<tr>
<td>COPYPEND</td>
<td>“COPY” on page 294</td>
</tr>
<tr>
<td>DATA</td>
<td>“DATA” on page 183</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>“DATACLAS” on page 423</td>
</tr>
<tr>
<td>DDTYPE</td>
<td>“DDTYPE” on page 397</td>
</tr>
<tr>
<td>DEFAULTIF</td>
<td>“fieldSpecification” on page 324</td>
</tr>
<tr>
<td>DELETEFILES</td>
<td>“DELETEFILES” on page 250</td>
</tr>
<tr>
<td>DISCARDDN</td>
<td>“DISCARDDN” on page 217</td>
</tr>
<tr>
<td>DISCARDS</td>
<td>“DISCARDS” on page 218</td>
</tr>
<tr>
<td>DRAIN_WAIT</td>
<td>“DRAIN_WAIT” on page 277</td>
</tr>
<tr>
<td>DSNPAT</td>
<td>“DSNPAT” on page 414</td>
</tr>
<tr>
<td>DSNTYPE</td>
<td>“DSNTYPE” on page 404</td>
</tr>
<tr>
<td>DSNUEXIT</td>
<td>“DSNUEXIT” on page 395</td>
</tr>
<tr>
<td>DSNUTILB</td>
<td>“DSNUTILB” on page 283</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>“EBCDIC” on page 203</td>
</tr>
<tr>
<td>ENFORCE</td>
<td>“ENFORCE” on page 263</td>
</tr>
<tr>
<td>ENUMROWS</td>
<td>“ENUMROWS” on page 229</td>
</tr>
<tr>
<td>ERRDDN</td>
<td>“ERRDDN” on page 216</td>
</tr>
<tr>
<td>EXCHANGE</td>
<td>“EXCHANGE” on page 256</td>
</tr>
<tr>
<td>EXPDT</td>
<td>“EXPDT” on page 420</td>
</tr>
<tr>
<td>FASTSWITCH</td>
<td>“FASTSWITCH” on page 253</td>
</tr>
<tr>
<td>FORCE</td>
<td>“FORCE” on page 279</td>
</tr>
<tr>
<td>FORCE_AT</td>
<td>“FORCE” on page 279</td>
</tr>
<tr>
<td>Command options</td>
<td>Reference</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>FORCE_DELAY</td>
<td>“FORCE” on page 279</td>
</tr>
<tr>
<td>FORCE_RPT</td>
<td>“FORCE” on page 279</td>
</tr>
<tr>
<td>FORMAT</td>
<td>“FORMAT” on page 196</td>
</tr>
<tr>
<td>GDGLIMIT</td>
<td>“GDGLIMIT” on page 422</td>
</tr>
<tr>
<td>IDCACHE</td>
<td>“IDCACHE” on page 273</td>
</tr>
<tr>
<td>IDCDDN</td>
<td>“IDCDDN” on page 249</td>
</tr>
<tr>
<td>IDENTITYOVERRIDE</td>
<td>“IDENTITYOVERRIDE” on page 276</td>
</tr>
<tr>
<td>IDERROR</td>
<td>“IDERROR” on page 275</td>
</tr>
<tr>
<td>IFALLOC</td>
<td>“IFALLOC” on page 400</td>
</tr>
<tr>
<td>IFDISCARDLIMIT</td>
<td>“IFDISCARDLIMIT” on page 224</td>
</tr>
<tr>
<td>IFDISCARDS</td>
<td>“IFDISCARDS” on page 223</td>
</tr>
<tr>
<td>IFZEROROWS</td>
<td>“IFZEROROWS” on page 225</td>
</tr>
<tr>
<td>IGNORE</td>
<td>“IGNORE” on page 221</td>
</tr>
<tr>
<td>IMPPLICIT_TZ</td>
<td>“IMPLICIT_TZ” on page 273</td>
</tr>
<tr>
<td>INDDN</td>
<td>“INDDN” on page 212</td>
</tr>
<tr>
<td>INDEX</td>
<td>“RESUME” on page 183</td>
</tr>
<tr>
<td>INDSN</td>
<td>“INDSN” on page 211</td>
</tr>
<tr>
<td>INLINE</td>
<td>“COPY” on page 294</td>
</tr>
<tr>
<td>INTO TABLE</td>
<td>“INTO TABLE” on page 306</td>
</tr>
<tr>
<td>KEEPDICTIOONARY</td>
<td>“KEEPDICTIOONARY” on page 188, “REPLACE” on page 309</td>
</tr>
<tr>
<td>LOAD</td>
<td>“LOAD” on page 183</td>
</tr>
<tr>
<td>LOADADDN</td>
<td>“LOADADDN” on page 213</td>
</tr>
<tr>
<td>LOADLOBDATA</td>
<td>“LOADLOBDATA” on page 286</td>
</tr>
<tr>
<td>LOBAVGPCET</td>
<td>“LOBAVGPCET” on page 287</td>
</tr>
<tr>
<td>LOG</td>
<td>“LOG” on page 270</td>
</tr>
<tr>
<td>LONGNAMETRUNC</td>
<td>“LONGNAMETRUNC” on page 202</td>
</tr>
<tr>
<td>MAPDDN</td>
<td>“MAPDDN” on page 284</td>
</tr>
<tr>
<td>MAXEXTSZ</td>
<td>“MAXEXTSZ” on page 402</td>
</tr>
<tr>
<td>MAXSORTMEMORY</td>
<td>“MAXSORTMEMORY” on page 242</td>
</tr>
<tr>
<td>MAXSORTS</td>
<td>“MAXSORTS” on page 236</td>
</tr>
<tr>
<td>MAXTAPE</td>
<td>“MAXTAPE” on page 394</td>
</tr>
</tbody>
</table>
### Command options

<table>
<thead>
<tr>
<th>Command options</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>MGMTCLAS</td>
<td>“MGMTCLAS” on page 424</td>
</tr>
<tr>
<td>MINSORTMEMORY</td>
<td>“MINSORTMEMORY” on page 241</td>
</tr>
<tr>
<td>NLPCTFREE</td>
<td>“NLPCTFREE” on page 257</td>
</tr>
<tr>
<td>NOSUBS</td>
<td>“NOSUBS” on page 205</td>
</tr>
<tr>
<td>NULLIF</td>
<td>“fieldSpecification” on page 324</td>
</tr>
<tr>
<td>ORDER</td>
<td>“ORDER” on page 225</td>
</tr>
<tr>
<td>ORIGINALDISP</td>
<td>“ORIGINALDISP” on page 252</td>
</tr>
<tr>
<td>PART</td>
<td>“PART” on page 310</td>
</tr>
<tr>
<td>PERIODOVERRIDE</td>
<td>“PERIODOVERRIDE” on page 309</td>
</tr>
<tr>
<td>POSITION</td>
<td>“fieldSpecification” on page 324</td>
</tr>
<tr>
<td>PREFORMAT</td>
<td>“PREFORMAT” on page 271 and “PART” on page 310</td>
</tr>
<tr>
<td>PRELOAD</td>
<td>“PRELOAD” on page 208</td>
</tr>
<tr>
<td>RECOVERYDDN</td>
<td>“RECOVERYDDN” on page 302</td>
</tr>
<tr>
<td>REDEFINE</td>
<td>“REDEFINE” on page 246</td>
</tr>
<tr>
<td>REGISTER</td>
<td>“COPY” on page 294</td>
</tr>
<tr>
<td>REPLACE</td>
<td>“REPLACE” on page 187 and “REPLACE” on page 309</td>
</tr>
<tr>
<td>REPORT</td>
<td>“REPORT” on page 222</td>
</tr>
<tr>
<td>RESUME</td>
<td>“RESUME” on page 183</td>
</tr>
<tr>
<td>RETPD</td>
<td>“RETPD” on page 421</td>
</tr>
<tr>
<td>RETRY</td>
<td>“RETRY” on page 279</td>
</tr>
<tr>
<td>RETRY_DELAY</td>
<td>“RETRY_DELAY” on page 279</td>
</tr>
<tr>
<td>REUSE</td>
<td>“REUSE” on page 249</td>
</tr>
<tr>
<td>SHORTMEMORY</td>
<td>“SHORTMEMORY” on page 239</td>
</tr>
<tr>
<td>SHRLEVEL</td>
<td>“SHRLEVEL” on page 189</td>
</tr>
<tr>
<td>SIZEPCT</td>
<td>“SIZEPCT” on page 410</td>
</tr>
<tr>
<td>SKIPFIELDS</td>
<td>“SKIPFIELDS” on page 304</td>
</tr>
<tr>
<td>SKIPIX</td>
<td>“SKIPIX” on page 206</td>
</tr>
<tr>
<td>SMS</td>
<td>“SMS” on page 401</td>
</tr>
<tr>
<td>SMSUNIT</td>
<td>“SMSUNIT” on page 402</td>
</tr>
<tr>
<td>SORTDEVT</td>
<td>“SORTDEVT” on page 234</td>
</tr>
<tr>
<td>Command options</td>
<td>Reference</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>SORTKEYS</td>
<td>“SORTKEYS” on page 207</td>
</tr>
<tr>
<td>SORTNUM</td>
<td>“SORTNUM” on page 235</td>
</tr>
<tr>
<td>SPACE</td>
<td>“SPACE” on page 411</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>“STORCLAS” on page 424</td>
</tr>
<tr>
<td>SWITCHTIME</td>
<td>“SWITCHTIME” on page 255</td>
</tr>
<tr>
<td>SYNC</td>
<td>“SYNC” on page 262</td>
</tr>
<tr>
<td>SYSDISC</td>
<td>“SYSDISC” on page 251</td>
</tr>
<tr>
<td>THRESHLD</td>
<td>“THRESHLD” on page 411</td>
</tr>
<tr>
<td>TOTALPAGEPCT</td>
<td>“TOTALPAGEPCT” on page 238</td>
</tr>
<tr>
<td>TRANSIDOVERRIDE</td>
<td>“TRANSIDOVERRIDE” on page 309</td>
</tr>
<tr>
<td>TSSAMPLEPCT</td>
<td>“TSSAMPLEPCT” on page 293</td>
</tr>
<tr>
<td>UNICODE</td>
<td>“UNICODE” on page 204</td>
</tr>
<tr>
<td>UNIQUECHECK</td>
<td>“UNIQUECHECK” on page 244</td>
</tr>
<tr>
<td>UNIQUEINTO</td>
<td>“UNIQUEINTO” on page 305</td>
</tr>
<tr>
<td>UNIT</td>
<td>“UNIT” on page 406</td>
</tr>
<tr>
<td>UNITCNT</td>
<td>“UNITCNT” on page 407</td>
</tr>
<tr>
<td>UPDATEDB2STATS</td>
<td>“UPDATEDB2STATS” on page 291</td>
</tr>
<tr>
<td>UPDATEMAXA</td>
<td>“UPDATEMAXA” on page 274</td>
</tr>
<tr>
<td>VOLCNT</td>
<td>“VOLCNT” on page 408</td>
</tr>
<tr>
<td>WHEN</td>
<td>“WHEN” on page 313</td>
</tr>
<tr>
<td>WORKDDN</td>
<td>“WORKDDN” on page 215</td>
</tr>
<tr>
<td>XBMID</td>
<td>“XBMID” on page 269</td>
</tr>
<tr>
<td>XMLAVGSIZE</td>
<td>“XMLAVGSIZE” on page 288</td>
</tr>
<tr>
<td>ZIIP</td>
<td>“ZIIP” on page 268</td>
</tr>
</tbody>
</table>

**LOADPLUS command syntax diagram**

The figures in this topic illustrates the syntax of the LOAD command.
Figure 8: LOADPLUS command syntax diagram (part 1 of 14)

LOAD
  DATA
  RESUME
  NO
  REPLACE
    KEEPDICTIONARY
      NO
      YES
    INDEX
      BUILD
      UPDATE
  SHRLLEVEL
    NONE
    REFERENCE
    CHANGE
    CHANGE block
    See CHANGE block detail
  FORMAT
    BMC
    BMCUNLOAD
    UNLOAD
    CSV
    CSV format block
    See CSV format block detail
  LONGNAMETRUNC
    MIDDLE
      BEGINNING
      END
  CCSID \textit{(sbc$s,\textit{mixed},\textit{dbcs})}
  NOSUBS
  SKIPPIX
    NO
    NUSIX
    SIX
  SORTKEYS \textit{integer}
    NO
  PRELOAD
    LOAD
    CONTINUE
    ANALYZE
    PAUSE
    ANYDISCARDS
    RETCODE
      integer

* Option has a corresponding installation option
Figure 9: LOADPLUS command syntax diagram (part 2 of 14)

* Option has a corresponding installation option
Figure 10: LOADPLUS command syntax diagram (part 3 of 14)

* Option has a corresponding installation option
Figure 11: LOADPLUS command syntax diagram (part 4 of 14)

- **ENFORCE**
  - **CONSTRAINTS**
    - **INFOR**
      - **CHECKPEND**
        - **YES**
        - **NO**
    - **NO**
- **ZIIP**
  - **ENABLED**
  - **DISABLED**
- **LOG**
  - **YES**
  - **NO**
  - **NOCOPYPEND**
- **PREFORMAT**
  - **NO**
  - **YES**
- **CENTURY** (ccyy, cyy)
- **IMPLIED_TZ** ‘timeZoneString’
- **IDCACHE**
  - **1000**
  - **integer**
- **UPDatemAXA**
  - **NO**
  - **YES**
- **IDERROR**
  - **DISCARD**
  - **FAIL**
- **IDENTITYOVERRIDE**
  - **NO**
  - **YES**
- **DRAIN_WAIT**
  - **NONE**
  - **UTIL**
  - **SQL**
  - **integer**
- **RETRY**
  - **255**
  - **integer**
- **RETRY_DELAY**
  - **1**
  - **integer**

* Option has a corresponding installation option
Figure 12: LOADPLUS command syntax diagram (part 5 of 14)

* Option has a corresponding installation option
Figure 13: LOADPLUS command syntax diagram (part 6 of 14)

**Statistics options**

- BMCSTATS
  - NO
  - YES
  - REPORT
- UPDATEDB2STATS
  - NO
  - YES
- TSSAMPLEPCT
  - 100
  - integer

**Copy options**

- COPY
  - NO
  - YES
- COPYPEND
  - YES
  - NO
- INLINE
  - YES
  - NO
- REGISTER
  - ALL
  - NONE
  - ddbname
- COPYDDN
  - BMCCPY
    - ddbname1
  - BMCCPZ
    - ddbname2
- RECOVERYDDN
  - BMCRCY
    - ddbname1
  - BMCRCZ
    - ddbname2

* Option has a corresponding installation option
Figure 14: LOADPLUS command syntax diagram (part 7 of 14)

Dynamic data set allocation options

CHANGE block detail – compatible only with LOAD RESUME YES

CSV format block detail

* Option has a corresponding installation option
Figure 15: LOADPLUS command syntax diagram (part 8 of 14)

**INTO TABLE block detail**

```
INTO TABLE tableCreator . tableName
```

- **REPLACE**
  - **PART** partitionNumber
  - **PERIOD OVERRIDE**
  - **TRANSID OVERRIDE**

- **REPLACE**
  - **KEEPDICTIONARY**
    - **NO**
    - **YES**

- **WHEN**
  - **condition block**
    - **TABLE=obid**
    - **AND**
      - **condition block**
    - **field specification block**

*Option has a corresponding installation option*
Figure 16: LOADPLUS command syntax diagram (part 9 of 14)

**Predicate block detail (RULES=STANDARD)**

```
predicate block detail (RULES=STANDARD)

fieldName  
columnName

(start    :end )

= 
< >
\~ =
IN
NOT IN

constant
```

**Predicate block detail (RULES=BMC)**

```
predicate block detail (RULES=BMC)

fieldName
columnName

(start    :end )

<
<=
=
< >
\~ =
\~ >

constant

>=

IN
NOT IN

constant
```

**Field specification block detail**

```
fieldName

POSITION (start    :end )

VALUE block
```

* Option has a corresponding installation option

---

Chapter 3  Syntax of the LOAD command  177
Figure 17: LOADPLUS command syntax diagram (part 10 of 14)

Field specification block detail – character and graphic data types

- CHAR
  - (length)
- VARCHAR
- BLOBF
- DBCLOBF
- CLOBF
- GRAPHIC
- VARGRAPHIC

Field specification block detail – numeric and binary data types

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
  - (length, scale)
  - ZONED
- FLOAT
- DECFLOAT
- BINARY
- VARBINARY
- BINARY VARYING

Field specification block detail – date, time, and timestamp data types

- DATE
- TIMESTAMP
  - (length)
  - CENTURY (ccyy, cyy)
- TIMESTAMP WITH TIME ZONE
- TIME
  - (length)
Figure 18: LOADPLUS command syntax diagram (part 11 of 14)

Field specification block detail – ROWID, LOB, XML, and exit data types

- ROWID
- BLOB
- CLOB
- DBCLOB
- XML
  - PRESERVE WHITESPACE
  - BINARYXML
- EXIT programName
  - PARM - ( constant )

Field specification block detail (continued)

- NULLIF
  - predicate block
    - ERROR
- DEFAULTIF
  - predicate block
    - ERROR
  - VALUE block
See VALUE block detail

VALUE block detail

- VALUE - ( constant )
  - CURRENT DATE
  - CURRENT TIME
  - CURRENT TIMESTAMP
See predicate block detail
Figure 19: LOADPLUS command syntax diagram (part 12 of 14)

DD type block detail

DDTYPE
  LOAD
    WORK
    SORTWORK
    DISCARD
    ERROR
    SYSMAP
    LOCPFCPY
    LOCBFCPY
    REMPFCPY
    REMBFCPY
    LOCPLCPY
    LOCBLCPY
    REMPLCPY
    REMBLCPY
    LOCPXCPY
    LOCBXCPY
    REMPXCPY
    REMBXCPY

  SMS
    NO
    YES

  SMSUNIT
    NO
    YES

MAXEXTSZ - ( - ( 0
  integer1,
  K
  TRK
  CYL
  )
  , - ( 0
  integer2,
  K
  TRK
  CYL
  )

* Option has a corresponding installation option
Figure 20: LOADPLUS command syntax diagram (part 13 of 14)

DD type block detail (continued)

```
DSNTYPE^a — (  NONE  )
               |       |
               v       v
   LARGE      BASIC      EXTREQ      EXTREFF
               |       |       |       |
               v       v       v       v
   NONE       LARGE      BASIC      EXTREQ      EXTREFF

UNIT^a — (  SYSALLDA  )
           |       |
           v       v
   unitName1      unitName2

 See SMS class block details

UNITCNT^a — (  0  )
            |       |
            v       v
   integer1      integer2

VOLCNT^a — (  25  )
            |       |
            v       v
   integer1      integer2

AVGVOLSP^a — (  -  )
             |       |
             v       v
   30000      TRK      CYL
             |       |
             v       v
   integer1      K      CYL

SIZEPCT^a — (  100  )
            |       |
            v       v
   primary      secondary

^ Option has a corresponding installation option
The rest of this chapter describes each LOADPLUS option.

The descriptions are in the order in which they appear in the syntax diagrams, which group the options according to the following functions:

- Basic processing options
- Statistics options
- Copy options
- Options that identify the tables and the data that you want to load
- Dynamic data set allocation options

**Basic LOADPLUS processing options**

The basic load processing options control most aspects of LOADPLUS execution.

**LOAD**

LOAD is the keyword for the LOADPLUS utility command.

**DATA**

DATA is available only for compatibility with the IBM DB2 LOAD utility command, and LOADPLUS treats this option as a comment.

With LOADPLUS, you identify the data to load by specifying the table name in the INTO TABLE option.

**RESUME**

RESUME tells LOADPLUS whether to add data to an empty table space or to a table space that already contains data.
In addition to interactions with other options (which are documented with those options), the value that you specify for the RESUME option can affect the following aspects of your load job:

- Performance ("Performance considerations for the RESUME and REPLACE command options" on page 664)
- Required data sets ("LOADPLUS DD statements" on page 432)
- Restart ("RESUME YES SHRLEVEL NONE" on page 475)

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

**NO**

RESUME NO (the default) tells LOADPLUS that the table space is empty, or that certain partitions are empty. LOADPLUS terminates when either of the following conditions exists:

- The table space contains data and you do not specify the REPLACE option.
- You specify PART n REPLACE and a partition that you are not replacing contains data.

**YES**

RESUME YES indicates that the table space might or might not be empty. LOADPLUS adds the new data to any existing data.

**Restriction**

You cannot use this option with LOAD REPLACE. If you want to replace only specific partitions of a table space, specify RESUME YES INTO TABLE with the PART n REPLACE option. If you want to replace only specific tables of a segmented table space, specify RESUME YES INTO TABLE with the REPLACE option.

**Additional considerations**

Note the following information about RESUME YES:

- Unless you specify INDEX UPDATE or SKIPIX, LOADPLUS unloads the existing indexes and merges them with the new index keys from the input data. LOADPLUS then sorts the merged index data and uses that data to build organized indexes during the LOAD phase or the COMBINED phase.
- LOADPLUS builds a new compression dictionary for the applicable table space or partition when all of the following conditions exist:
— You are loading an empty partitioned table space or partition.
— The table space is defined with the COMPRESS YES attribute.
— A compression dictionary does not currently exist for the table space or partition.

**Note**
For a partition-by-growth table space, if the first partition has a compression dictionary and any of the subsequent partitions are empty, LOADPLUS copies the dictionary from the first partition instead of building a new dictionary.

For all other LOAD RESUME YES jobs, LOADPLUS keeps any existing compression dictionary.

**Note**
If the existing dictionary is unusable, LOADPLUS continues but does not compress any rows.

**INDEX**

This option tells LOADPLUS whether to build or update an index when specifying RESUME YES.

**Restrictions**

The following restrictions apply to the INDEX option:

- LOADPLUS does not build or update indexes when either of the following conditions exists:
  - You are running an SQLAPPLY load.
  - LOADPLUS is invoking DSNUTILB.

- If you specify SKIPIX SIX or SKIPIX NUSIX, LOADPLUS does not build or update the indexes that you are skipping.

**BUILD**

This option, which is the default, tells LOADPLUS to unload the existing indexes and merge them with the new index keys from the input data. LOADPLUS then sorts the merged index data and uses that data to build organized indexes during the LOAD phase or the COMBINED phase.

**UPDATE**

This option tells LOADPLUS to add index entries to the existing indexes. With INDEX UPDATE, LOADPLUS does not reorganize indexes as it does when INDEX BUILD is in effect.

**Requirement**
When you specify INDEX UPDATE for a two-phase load, you must also specify UNIQUECHECK NO. If you specify UNIQUECHECK YES or CLUSTER, LOADPLUS fails and displays message BMC51419E.

**Restrictions**

The following restrictions apply to INDEX UPDATE:

- LOADPLUS terminates if you are loading XML or LOB data.
- LOADPLUS changes INDEX UPDATE to INDEX BUILD when you specify TABLE `tableName` REPLACE.
- LOADPLUS invokes DSNUTILB when loading a temporal table and the following conditions also exist:
  - A participating index is defined with BUSINESS_TIME WITHOUT OVERLAPS.
  - Any SHRLEVEL value is in effect except SHRLEVEL CHANGE SQLAPPLY.

**Additional considerations**

The following considerations apply to the INDEX UPDATE option:

- BMC recommends that you limit the use of UPDATE to those cases where you are adding a small percentage of the total amount of existing data. If you are adding a large percentage, using UPDATE can impact optimal performance of the SQL that uses the index in processing.
- If you specify PART `n` REPLACE (unless you are replacing all partitions), LOADPLUS rebuilds the data-sorting indexes and updates the participating non-data-sorting indexes and nonpartitioned data-sorting indexes.
- When you specify INDEX UPDATE, LOADPLUS does not unload existing index records, so LOADPLUS cannot check for uniqueness in the PRELOAD phase. If duplicates exist, LOADPLUS detects them in the LOAD or COMBINED phase, loads the table space, deletes the duplicate records, and issues messages BMC50258E and BMC51477I. For a two-phase load, LOADPLUS does not write these duplicates to the discard data set and they do not count toward any limit that you specify with the DISCARDS option.
- When you specify INDEX UPDATE and a failure occurs during index update processing, you will need to recover your table space and...
indexes. If you attempt to restart, LOADPLUS terminates and issues message BMC51435S.

- When you specify INDEX UPDATE and index update processing completes, but the job fails in the LOAD phase, you must restart the job by specifying RESTART without PHASE. If you specify RESTART(PHASE), LOADPLUS terminates and issues message BMC51436E.

REPLACE

REPLACE tells LOADPLUS to replace any existing data in the table space (not only the data in the tables or partitions that you specify in the INTO TABLE option).

LOADPLUS deletes the existing data in the table space before loading.

LOADPLUS converts data that is in basic row format to reordered row format. However, if the table contains a VALIDPROC or EDITPROC, LOADPLUS loads the data according to the existing row format on the table space.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

Restrictions

LOAD REPLACE has the following restrictions:

- If you specify the REPLACE option, you cannot specify the PART n REPLACE or TABLE tableName REPLACE option.

- LOADPLUS terminates if you specify REPLACE for a system-period temporal table that is enabled for versioning.
KEE PDIC T I O N A R Y

This option tells LOADPLUS whether to keep the existing compression dictionary. If you specify KEE PDIC T I O N A R Y without a value, LOADPLUS assumes KEE PDIC T I O N A R Y YES. For details about when LOADPLUS builds and keeps a dictionary, as well as additional considerations when using compression, see “Considerations for table space compression” on page 140.

LOADPLUS also provides KEE PDIC T I O N A R Y support at the partition level. For information about how to use the PART REPLACE options, see “PART” on page 310.

Restrictions

The following restrictions apply to the KEE PDIC T I O N A R Y option:

- The KEE PDIC T I O N A R Y option is valid only if the table space that you are loading has the COMPRESS YES attribute.

- If a table space is compressed and a load job would convert the row format from BRF to RRF, LOADPLUS builds a new dictionary except when both of the following options are in effect:

  - The value of the LOADPLUS KEE PDIC T I O N A R Y option is YES.

  - The value of the DB2 subsystem parameter HONOR_KEE PDIC T I O N A R Y is YES.

Specifying the default

You can specify the default for the KEE PDIC T I O N A R Y option in your installation options module by using the KEE PDIC T I O N A R Y installation option. LOADPLUS was shipped with a default value of NO for this option. The command option overrides the default that is in the installation options module.

YES

If you specify KEE PDIC T I O N A R Y YES, LOADPLUS keeps the existing compression dictionary. If a dictionary already exists, LOADPLUS uses it for compression. If a dictionary does not exist, LOADPLUS builds one in the PRELOAD phase or the COMBINED phase. After completely building the dictionary, LOADPLUS compresses the data.

When invoking DSNUTILB,

- For an SQLAPPLY load, LOADPLUS ignores this option.
- Otherwise, LOADPLUS passes the value of this option to the IBM DB2 LOAD utility as KEEPDICTIONARY.

**NO**

If you specify KEEPDICTIONARY NO, LOADPLUS builds a new compression dictionary. LOADPLUS builds the dictionary in the PRELOAD phase or the COMBINED phase. After completely building the dictionary, LOADPLUS compresses the data.

**Restriction**

When invoking DSNUTILB, LOADPLUS ignores this option.

**SHRLEVEL**

SHRLEVEL specifies the level of access that DB2 has to the objects that you are loading during LOADPLUS processing.

**CHANGE block detail — compatible only with LOAD RESUME YES**

See “**SHRLEVEL considerations in LOADPLUS**” on page 87 for important information about using this option.

**Specifying the default**
You can specify the default for the SHRLEVEL option in your installation options module by using the SHRLEVEL installation option. LOADPLUS was shipped with a default value of NONE for this option. The command option overrides the default that is in the installation options module.

**NONE**

SHRLEVEL NONE tells LOADPLUS to stop the objects that you are loading, making them unavailable during the entire load process.

**REFERENCE**

SHRLEVEL REFERENCE tells LOADPLUS to make the objects that you are loading available in read-only status. LOADPLUS writes the loaded data to staging data sets. The original VSAM data sets remain intact throughout the load process. This nondestructive process allows you to easily restart from a failure or make the objects available without having to recover.

For more information about using this option, see “Running SHRLEVEL REFERENCE” on page 89.

**LOAD RESUME YES**

For LOAD RESUME YES, LOADPLUS uses the Instant Snapshot functionality of XBM or SUF. You must have XBM or SUF installed and available to LOADPLUS. For more information, see “XBM and SUF considerations in LOADPLUS” on page 160 and “Running SHRLEVEL REFERENCE” on page 89.

**Restriction**

This option is not valid when LOADPLUS invokes DSNUTILB. LOADPLUS issues message BMC50178E and terminates.

**CHANGE**

SHRLEVEL CHANGE allows the objects that you are loading to remain in read/write status. This option functions differently depending on whether you specify LOAD REPLACE, LOAD RESUME YES with PART n REPLACE, or LOAD RESUME YES without PART n REPLACE.

**With LOAD REPLACE or LOAD RESUME YES PART n REPLACE**

For SHRLEVEL CHANGE with either LOAD REPLACE or LOAD RESUME YES PART n REPLACE, LOADPLUS writes the loaded data to staging data sets. The original VSAM data sets remain intact throughout the load process. This nondestructive process allows you to easily restart from a failure or make the objects available without having to recover.
WARNING
This option fully replaces the data in the table space or partition, including any inserts, updates, or deletes to the base object that occur during the load process.

For more information about using this option, including the procedure and restrictions for running this type of load job, see “Running LOAD REPLACE or LOAD RESUME YES PART n REPLACE” on page 91.

This option is not valid when LOADPLUS invokes DSNUTILB. LOADPLUS issues message BMC50178E and terminates.

With LOAD RESUME YES without PART n REPLACE

For SHRLEVEL CHANGE with LOAD RESUME YES without PART n REPLACE, the object that you are loading remains in read/write status while LOADPLUS loads the data by using SQL insert processing. For this type of load, LOADPLUS uses the High-speed Apply Engine component of the BMC Log Master for DB2 product. This feature does not require that you have a license for the Log Master product. However, the High-speed Apply component must be installed and made available through the STEPLIB, JOBLIB, or LINKLIST at runtime.

For important information about using this option, including the procedure for running this type of load and compatibility with other LOAD command options, see “Running LOAD RESUME YES (SQLAPPLY load)” on page 93.

Note
For ease of reference, this book refers to this type of load as an SQLAPPLY load.

If you specify SHRLEVEL CHANGE with LOAD RESUME YES, the following additional command options allow you to control your load job.

SQLAPPLY

This option is valid only when you specify LOAD RESUME YES without PART REPLACE.

SQLAPPLY specifies that you want to use the type of online load that uses High-speed Apply. For this type of load, LOADPLUS passes processing to High-speed Apply during the COMBINED phase. For considerations, including restrictions, when using this option, see “SHRLEVEL CHANGE considerations” on page 91.

You can specify one or more of the following options to further control your SQLAPPLY load job. The values for these options are passed to High-speed Apply as configuration parameters for use during High-speed Apply processing.
**Additional considerations**

The following information applies to the SQLAPPLY options:

- When you specify more than one of the following SQLAPPLY options, your syntax must follow the sequence that is shown in the syntax diagram. For example, if you specify APCOMMIT, it must always appear before APOWNER.

- When invoking DSNUTILB, LOADPLUS ignores these options.

**APMAXAGENTS**

APMAXAGENTS defines the maximum number of agents that High-speed Apply can start for this load. Valid values for this option are 1 through the number of batch threads that are available in your DB2 subsystem. APMAXAGENTS affects only partitioned or multi-table table spaces. LOADPLUS uses at most one agent per partition and table. For partition-by-growth table spaces, LOADPLUS uses only one agent per load regardless of the value that you specify.

LOADPLUS passes the value of this parameter to the MaxAgents parameter in the Agent section of High-speed Apply. For more information about the MaxAgents parameter, see the *High-speed Apply Engine Reference Manual*.

**Note**

If you specify a value greater than 1 when the following conditions exist, you might encounter a task deadlock, which could result in the LOADPLUS job failing:

- You are loading a partitioned (except partition-by-growth) table space.

- The table space is defined with LOCKSIZE TABLESPACE or LOCKSIZE ANY.

If you encounter this deadlock condition, BMC recommends that you change the value of this option to 1.

**Specifying the default**

You can specify the default for the APMAXAGENTS option in your installation options module by using the APMXAGNT installation option. LOADPLUS was shipped with a default value of 10 for this option. The command option overrides the default that is in the installation options module.
APCOMMIT

APCOMMIT specifies the number of records that each High-speed Apply agent should load before issuing a COMMIT statement. Valid values for this option are 1 through 32767.

LOADPLUS passes the value of this parameter to the StatementCount parameter in the CommitTriggers section of High-speed Apply. For more information about the StatementCount parameter, see the High-speed Apply Engine Reference Manual.

**Specifying the default**

You can specify the default for the APCOMMIT command option in your installation options module by using the APCOMMIT installation option. LOADPLUS was shipped with a default value of 2500 for this option. The command option overrides the default that is in the installation options module.

APRETRYLIM

APRETRYLIM tells High-speed Apply what measure to use to determine the limit for retry attempts following SQL -911, -913, or -904 errors. The value for this option is used in conjunction with the APRETRYVAL option. The following table describes valid values for the APRETRYLIM option:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COUNT</td>
<td>COUNT tells High-speed Apply to base the limit for these retry attempts on the number of attempts that the APRETRYVAL option specifies.</td>
</tr>
<tr>
<td>TIME</td>
<td>TIME tells High-speed Apply to retry until it reaches the number of seconds that the APRETRYVAL option specifies.</td>
</tr>
</tbody>
</table>

LOADPLUS passes the value of this parameter to the RetryLimit parameter in the Conflict section of High-speed Apply. For more information about the RetryLimit parameter, see the High-speed Apply Engine Reference Manual.

**Specifying the default**

You can specify the default for the APRETRYLIM command option in your installation options module by using the APRETLIM installation option. LOADPLUS was shipped with a default value of COUNT for this option. The command option overrides the default that is in the installation options module.
APRETRYVAL

APRETRYVAL provides High-speed Apply with the retry limit for each unit of recovery in the load job. The value for this option is used in conjunction with the APRETRYLIM option. Valid values are 0 through 32767. A value of 0 tells High-speed Apply not to retry.

LOADPLUS passes the value of this parameter to the RetryValue parameter in the Conflict section of High-speed Apply. For more information about the RetryValue parameter, see the High-speed Apply Engine Reference Manual.

Specifying the default

You can specify the default for the APRETRYVAL command option in your installation options module by using the APRETVAL installation option. LOADPLUS was shipped with a default value of 5 for this option. The command option overrides the default that is in the installation options module.

APCOLLECTION

APCOLLECTION provides High-speed Apply with the ID for the collection to which High-speed Apply dynamically binds packages during execution. You can specify a collection ID with a length of up to 128 bytes. You cannot specify a null value for this option.

LOADPLUS passes this value to High-speed Apply. If you do not specify a value for this command option, one of the following actions occurs:

- If the value of the APCOLLECTION installation option is not null, LOADPLUS passes the value to High-speed Apply.

- If the value of the APCOLLECTION installation option is null, LOADPLUS does not pass any value to High-speed Apply. High-speed Apply uses the default value of the CollectionID parameter in the Bind section of High-speed Apply.

For more information, see the High-speed Apply Engine Reference Manual.

Specifying the default

You can specify the default for the APCOLLECTION command option in your installation options module by using the APCOLLECTION installation option. LOADPLUS was shipped without a value for this option. The command option overrides the default that is in the installation options module.
APOWNER

APOWNER provides High-speed Apply with the authorization ID to use to bind the DB2 plan and packages for the apply request. You can specify an owner ID with a length of up to 128 bytes. You cannot specify a null value for this option.

LOADPLUS passes this value to High-speed Apply. If you do not specify a value for this command option, one of the following actions occurs:

- If the value of the APOWNER installation option is not null, LOADPLUS passes the value to High-speed Apply.
- If the value of the APOWNER installation option is null, LOADPLUS does not pass any value to High-speed Apply. High-speed Apply uses the default value of the BindOwner parameter in the Bind section of High-speed Apply.

For more information, see the High-speed Apply Engine Reference Manual.

Specifying the default

You can specify the default for the APOWNER command option in your installation options module by using the APOWNER installation option. LOADPLUS was shipped without a value for this option. The command option overrides the default that is in the installation options module.

APMULTIROW

APMULTIROW tells High-speed Apply whether to insert multiple rows in a single insert action and, if so, how many rows to insert at one time. You can specify a value from 0 through 32767. A value of 0 or 1 tells High-speed Apply to insert one row at a time.

LOADPLUS passes this value to the MultiRowInsert and MaxRows parameters in the MultiRowInsert section of High-speed Apply. For more information about these parameters, see the High-speed Apply Engine Reference Manual.

Specifying the default

You can specify the default for the APMULTIROW command option in your installation options module by using the APMULTIROW installation option. LOADPLUS was shipped with a default value of 100 for this option. The command option overrides the default that is in the installation options module.
FORMAT

FORMAT specifies the format of the input data in the SYSREC data set when your input data is in certain formats.

Specify this option when your input data is in one of the following formats:

- Format generated by REORG PLUS by creating an archive data set (SYSARC) with ARCHFORMAT BMC (FORMAT BMC)
- Format generated by UNLOAD PLUS by using the FORMAT BMCLOAD option (FORMAT BMCUNLOAD)
- Format generated by one of the following methods (FORMAT UNLOAD):
  - The IBM DB2 REORG utility by using the UNLOAD ONLY option
  - REORG PLUS by creating an archive data set (SYSARC) with ARCHFORMAT DB2
- Comma-separated-value (CSV) format (FORMAT CSV)
- Spanned-record format (FORMAT SPANNED)
BMC

BMC specifies that the input data in the SYSREC data set is in the format that was generated by REORG PLUS by using the SYSARC data set and the ARCHFORMAT BMC option. The SYSARC data set is an archive data set that contains rows that are discarded during the reorganization.

Restrictions

Note the following restrictions for FORMAT BMC:

■ LOADPLUS terminates when either of the following conditions exists:
  — You are loading LOB or XML data.
  — You also specify CONTINUEIF.

■ The table that you are loading must have the same definition as the table from which the archive rows originated. LOADPLUS loads the records that match the table OBID of the table that you specify in the INTO TABLE option or the input table OBID that you specify in the WHEN TABLE=obid option of the LOAD command. If the table definitions are different, results of the load job are unpredictable.

  **Note**
  When you specify WHEN TABLE=obid, the OBID must be a decimal number. For more considerations when using this option, see “WHEN” on page 313.

■ If you specify FORMAT BMC, you cannot include field specifications on your LOAD command.

■ LOADPLUS allows you to specify a WHEN condition with the FORMAT BMC option. However, the condition must reference columns of the table only; it cannot reference a (start:end) field. See “WHEN” on page 313 for more information.

■ This option is not valid when LOADPLUS invokes DSNUTILB. LOADPLUS issues message BMC50178E and terminates.

BMCUNLOAD

BMCUNLOAD specifies that the input data in the SYSREC data set is in an internal format that was generated by UNLOAD PLUS. This input data is data that was unloaded by using the UNLOAD PLUS option FORMAT BMCLOAD. For information about how to unload data in this format, see the *UNLOAD PLUS for DB2 Reference Manual*.

LOADPLUS loads the records from the input data set that match the table object identifier (OBID) that you specify in the WHEN TABLE=obid option of the LOAD command.
command. In this case, the OBID is the object identifier that UNLOAD PLUS assigned when unloading the data, not the table’s DB2 object identifier. The LOAD control cards that UNLOAD PLUS generated contain this OBID. For more information about specifying this option with FORMAT BMCUNLOAD, see “WHEN” on page 313.

BMC strongly recommends that you use the LOAD control cards that UNLOAD PLUS generated.

FORMAT BMCUNLOAD is useful for migrating data to duplicate tables or from development to production databases. This option provides an improved performance benefit because the data is never converted to an external format and because LOADPLUS does not need to perform data verification.

For more information, see the following references:

- For information about using the FORMAT BMCUNLOAD option, see “Using data from UNLOAD PLUS in internal BMC format” on page 132.

- For an example that uses this option, see “Example 14 LOAD REPLACE from UNLOAD PLUS data in internal format” on page 588.

**Restrictions**

Note the following restrictions for FORMAT BMCUNLOAD:

- LOADPLUS terminates when either of the following conditions exists:
  - You are loading LOB or XML data.
  - You also specify CONTINUEIF.

- If you specify FORMAT BMCUNLOAD, you cannot include field specifications on your LOAD command.

- All partitions in the table space that you are loading must be in the same row format (basic or reordered).

- With a few exceptions, the table that you are loading must have the same definition as the table that was unloaded with UNLOAD PLUS. For details, see “Table structure considerations for FORMAT BMCUNLOAD” on page 134.

- This option is not valid when LOADPLUS invokes DSNUTILB. LOADPLUS issues message BMC50178E and terminates.
UNLOAD

This option specifies that the input data in the SYSREC data set is in the format that was generated in one of the following ways:

- By the IBM DB2 REORG utility by using the UNLOAD ONLY option
- By REORG PLUS by using a SYSARC data set and specifying ARCHFORMAT

LOADPLUS loads the records from the input data set that match the OBID of the table that you specify in the INTO TABLE option or the input table OBID that you specify in the WHEN TABLE=obid option with FORMAT UNLOAD. Unlike the IBM DB2 LOAD utility, LOADPLUS allows you to specify a WHEN condition with the FORMAT UNLOAD option.

Restrictions

Note the following restrictions for FORMAT UNLOAD:

- LOADPLUS terminates when either of the following conditions exists:
  — You are loading LOB or XML data.
  — You also specify CONTINUEIF.

- If you specify FORMAT UNLOAD, you cannot include any field specifications on your LOAD command.

- If you specify a WHEN condition, the condition must reference columns of the table only; it cannot reference a (start:end) field. For more information, see “WHEN” on page 313.

- If you are loading a SYSARC file from REORG PLUS, the table that you are loading must have the same definition as the table from which the archive rows originated. LOADPLUS loads the records that match one of the following table OBIDs:
  — Table OBID of the table that you specify in the INTO TABLE option
  — Input table OBID that you specify in the WHEN TABLE=obid option

  If the table definitions are different, results of the load job are unpredictable.

  Note

  When you specify WHEN TABLE=obid, the OBID must be a decimal number. For more considerations when using this option, see “WHEN” on page 313.

CSV

The CSV option allows you to load data that has been unloaded in comma-separated-value (CSV) format. This data can originate from any product that
generates CSV output, including distributed databases, spreadsheet applications, and UNLOAD PLUS. In CSV-formatted data, a particular character separates each field from other fields. Also, in most cases, a particular pair of characters encloses each non-numeric field.

**Note**

CSV files are also referred to as comma-delimited or delimited files.

To use this option, your data and LOAD command must meet the requirements that are specified in “CSV data” on page 126.

When you use more than one of the following CSV options, your syntax must follow the sequence that is shown in the syntax diagram. For example, if you specify TERMINATEDBY, it must always appear before ENCLOSEDBY.

**Restrictions**

LOADPLUS has the following restrictions on FORMAT CSV:

- LOADPLUS terminates when either of the following conditions exists:
  - You are loading LOB or XML data.
  - You also specify CONTINUEIF.

- When your input data is Unicode, LOADPLUS assumes that the values for the TERMINATEDBY, ENCLOSEDBY, and AND options are in EBCDIC format. LOADPLUS translates the values, if necessary, to the CCSID of the input data. Note the following restrictions for these options:
  - Do not specify values for these options that would expand to multiple-byte characters when they are translated.
  - If you specify a hexadecimal value, it must be in the internal format of the input, and the length must be no more than one byte.

**TERMINATEDBY**

This option specifies the character that your input data uses to delimit fields of data. The default is a comma. You cannot specify a null value for TERMINATEDBY.

**ENCLOSEDBY**

The ENCLOSEDBY option specifies the character that your input data uses on the left side to enclose fields of data. The default is a double quotation mark ("). You cannot specify a null value for ENCLOSEDBY.
For each field, if LOADPLUS does not find the ENCLOSEDBY character that you specify or default to, LOADPLUS assumes that the field is not enclosed by any character.

**AND**

This option specifies the character that your input data uses on the right side to enclose fields of data. You cannot specify a null value for AND.

If you do not specify a value for this option, LOADPLUS assumes that your input data uses the value that you specified for the ENCLOSEDBY option. If you did not specify a value for the ENCLOSEDBY option, LOADPLUS assumes that your input fields are enclosed by double quotation marks or are not enclosed by any character.

**SPANNE**

The SPANNED option indicates whether you are loading LOB or XML data from a variable-block spanned (VBS) data set that is in spanned-record format.

**NO**

SPANNED NO, the default, indicates that the input data set is not in spanned-record format.

If your input is in spanned-record format and FORMAT SPANNED NO is in effect, results are unpredictable.

**YES**

SPANNED YES indicates that the input data set is in spanned record format. The following considerations apply when you specify SPANNED YES:

**WARNING**

If you load a data set in spanned-record format that was created by any product other than UNLOAD PLUS, results are unpredictable. For example, LOADPLUS might appear to complete normally but not load the data correctly.

- LOADPLUS terminates when any of the following conditions exists:
  - You specify FORMAT SPANNED YES, but your input data set is not allocated as RECFM=VBS with LRECL=32768.
  - You specify one of the following options:
    - CONTINUEIF
    - PRELOAD CONTINUE (two-phase load)
    - LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY
Your data or CCSID specification requires that LOADPLUS translate the data.

You are running a DSNUTILB load.

If you specify ENUMROWS AUTO, LOADPLUS cannot make accurate estimates when your input data is in spanned-record format. When you are loading from an input data set that is in spanned-record format, BMC recommends that you specify values for the ENUMROWS option.

When loading from an input data set that is in spanned-record format, LOADPLUS writes any discarded records to a discard data set that is in spanned-record format. If you have allocated a discard data set in your JCL, LOADPLUS overwrites the allocation parameters to RECFM=VBS, LRECL=32768.

To load this data set, you must specify FORMAT SPANNED YES on your LOAD command.

LONGNAMETRUNC

LONGNAMETRUNC specifies where to truncate names that are longer than the area that is available in report-style messages.

- LONGNAMETRUNC
- MIDDLE
- BEGINNING
- END

Restriction

When invoking DSNUTILB, the utility ignores this option.

Specifying the default

You can specify the default for the LONGNAMETRUNC command option in your installation options module by using the LONGNAMETRUNC installation option. The utility was shipped with a default value of MIDDLE for this option. The command option overrides the default value that is in the installation options module.
MIDDLE

MIDDLE specifies to truncate outward from the middle of the name. You can shorten this value to M. The symbols >< indicate the truncated section, as shown in the following example:

```
BMC50521I CREATOR          NAME
BMC50522I LONG_CR><OR_NAME VERY_LONG_TABLE_NAME_><HAN_THIRTY_CHARACTERS
```

BEGINNING

BEGINNING, or B, specifies to truncate from the beginning of the name. The symbols << indicate the truncated section, as shown in the following example:

```
BMC50521I CREATOR          NAME
BMC50522I <<G_CREATOR_NAME <<_TABLE_NAME_GREATER_THAN_THIRTY_CHARACTERS
```

END

END, or E, specifies to truncate from the end of the name. The symbols >> indicate the truncated section, as shown in the following example:

```
BMC50521I CREATOR          NAME
BMC50522I LONG_CREATOR_N>> VERY_LONG_TABLE_NAME_GREATER_THAN_THIRTY_C>>
```

EBCDIC

EBCDIC specifies that the input data is encoded in EBCDIC using the DB2 installation default coded character set identifiers (CCSIDs).

EBCDIC is the default. For information about data translation, see “Data translation” on page 383.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

Restriction

LOADPLUS ignores this option if you also specify FORMAT BMCUNLOAD.
**ASCII**

The ASCII option specifies that the input data is encoded in ASCII using the DB2 installation default CCSIDs.

For information about data translation, see “Data translation” on page 383.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

*Restriction*

LOADPLUS ignores this option if you also specify FORMAT BMCUNLOAD.

**UNICODE**

This option specifies that the data that you are loading is in Unicode format.

For information about data translation, see “Data translation” on page 383.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

*Restriction*

LOADPLUS ignores this option if you also specify FORMAT BMCUNLOAD.

**CCSID**

This option allows you to specify up to three CCSIDs for use in encoding input data.
The three CCSIDs are for single-byte character set (SBCS), MIXED character set, and double-byte character set (DBCS) data, respectively. If you omit a CCSID value or specify 0, LOADPLUS uses the corresponding DB2 system default CCSID for the encoding scheme that you specify or default to (EBCDIC, ASCII, or UNICODE). For more details, including supported translations, see “Data translation” on page 383.

**Note**
BMC recommends that you use this option only to override your DB2 system default CCSID values.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

**Restrictions**

Note the following restrictions:

- LOADPLUS ignores this option if you also specify FORMAT BMCUNLOAD.

- LOADPLUS supports translation from one CCSID to another except translations that require the use of a conversion procedure.

**NOSUBS**

NOSUBS tells LOADPLUS not to accept substitution characters during translation between CCSIDs.

If you specify NOSUBS and LOADPLUS encounters a record that requires substitution, LOADPLUS does not load the record, but writes it to your discard file (SYSDISC).

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

**Restriction**

LOADPLUS ignores this option if you also specify FORMAT BMCUNLOAD.
**SKIPIX**

The SKIPIX option tells LOADPLUS whether to skip building or updating the nonpartitioned indexes that are associated with the partitioned table space that you are loading.

Skipped indexes do not participate in any aspect of the load job, including checking for duplicate keys.

**WARNING**

BMC strongly recommends that you do not specify SKIPIX SIX to ignore unique nonpartitioned indexes if there is any possibility of loading duplicate values. If running with this option results in loading duplicate values, follow your standard procedures for eliminating duplicates when creating a unique index.

**Restrictions**

If you specify NUSIX or SIX under any of the following conditions, LOADPLUS terminates:

- You are loading a nonpartitioned table space.
- You are running an SQLAPPLY load job.
- You are skipping a clustering index and you also specify ORDER YES.
- LOADPLUS invokes DSNUTILB.

**Additional considerations**

If you specify NUSIX or SIX, consider the following information:

- LOADPLUS ignores these options for the following XML and LOB indexes:
  - Document ID
  - Node ID
  - Row ID
  - Auxiliary (on the LOB table space)

- The nonparticipating indexes can be in any restrictive status, such as CHECK pending (CHKP), unless they were created as DEFINE NO.
With the following exception, LOADPLUS sets the nonparticipating indexes to RBDP, RBDP*, or PSRBD status and completes with return code 4.

**Note**

If you specify RESUME YES (without PART n REPLACE) and no rows are loaded, LOADPLUS leaves the nonparticipating indexes in the same status that they were in at the start of the load job.

In those cases where LOADPLUS sets the nonparticipating indexes to a rebuild pending status, you must take appropriate action to remove the pending status (for example, rebuilding the index) following the load job.

**NO**

SKIPIX NO (the default) tells LOADPLUS to build or update all nonpartitioned indexes that are associated with the table that you are loading.

**NUSIX**

SKIPIX NUSIX tells LOADPLUS to not build or update any non-unique nonpartitioned indexes that are associated with the table that you are loading.

**SIX**

SKIPIX SIX tells LOADPLUS to not build or update any nonpartitioned indexes, unique or non-unique, that are associated with the table that you are loading.

**SORTKEYS**

When LOADPLUS invokes DSNUTILB, the SORTKEYS option specifies that index keys are to be sorted in parallel.

\[ \text{SORTKEYS} \quad \text{integer} \quad \text{NO} \]

This option is meaningful only when LOADPLUS invokes DSNUTILB. Otherwise, LOADPLUS treats this option as a comment.

**integer**

An integer value specifies the estimated number of index keys to be sorted.
NO

SORTKEYS NO indicates that keys are not to be sorted in parallel.

PRELOAD

The PRELOAD option controls the execution of LOADPLUS.

Specifying the default

You can specify the default for the PRELOAD command option in your installation options module by using the PRELOAD installation option. However, you can specify only CONTINUE or LOAD for the installation option. LOADPLUS was shipped with a default value of LOAD for this option. Any PRELOAD command option that you specify overrides the default that is in the installation options module.

LOAD

If you specify PRELOAD LOAD, LOADPLUS uses single-phase load processing, which combines the functions of the PRELOAD and LOAD phases into the COMBINED phase. LOADPLUS reads the input data and copies it directly to the table space. This phase can improve performance in most cases. See “Performance considerations for the PRELOAD LOAD installation and command options” on page 663 for details.

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.
CONTINUE

PRELOAD CONTINUE tells LOADPLUS to use two-phase processing and to continue with the LOAD phase after the PRELOAD phase is completed.

Restrictions

The following restrictions apply to PRELOAD CONTINUE:

- If you are loading LOB or XML data, LOADPLUS invokes DSNUTILB.
- If you also specify ENFORCE RI or ENFORCE CONSTRAINTS, LOADPLUS terminates. If you want LOADPLUS to check referential constraints during the load job, specify PRELOAD LOAD.
- LOADPLUS ignores this option when either of the following conditions exists:
  — LOADPLUS is invoking DSNUTILB.
  — You are running an SQLAPPLY load.

ANALYZE

The PRELOAD ANALYZE option tells LOADPLUS to use two-phase processing, but to stop after the optimization phase of the PRELOAD phase. No data is read and no tables are affected. LOADPLUS displays message BMC51496I indicating the optimal number of sort tasks, read tasks, and index tasks. You can use this information to adjust the number of data sets that you specify in your JCL before resubmitting your job.

Restrictions

LOADPLUS terminates when any of the following conditions exists and you specify PRELOAD ANALYZE:

- You are loading LOB or XML data.
- You are running an SQLAPPLY load.
- LOADPLUS is invoking DSNUTILB.
- You also specify ENFORCE RI or ENFORCE CONSTRAINTS.
  If you want LOADPLUS to check referential constraints during the load job, specify PRELOAD LOAD.

PAUSE

The PRELOAD PAUSE option tells LOADPLUS to use two-phase processing, but to pause after the PRELOAD phase.
This option is particularly useful if you want to control your outage when running LOAD REPLACE with SHRLEVEL NONE. For this type of load, LOADPLUS normally stops the table space during the LOAD phase instead of the PRELOAD phase. By specifying PRELOAD PAUSE, you can schedule the LOAD phase to minimize the impact on data availability.

**Restarting**

You can restart the utility at the beginning of the LOAD phase by specifying RESTART or RESTART(PHASE) on your EXEC statement. Note the following information about restarting your PRELOAD PAUSE job:

- You do not need to change the PRELOAD option on the LOAD command. When restarting a PRELOAD PAUSE job, LOADPLUS ignores the PAUSE specification.

- If you also specify LOAD RESUME YES and SHRLEVEL NONE is in effect, the participating objects must remain in stopped status before you restart your job.

**Restrictions**

LOADPLUS terminates when any of the following conditions exists and you specify PRELOAD PAUSE:

- You are loading LOB or XML data.
- You are running an SQLAPPLY load.
- LOADPLUS is invoking DSNUTILB.
- You also specify ENFORCE RI or ENFORCE CONSTRAINTS.

If you want LOADPLUS to check referential constraints during the load job, specify PRELOAD LOAD.

**ANYDISCARDS**

The ANYDISCARDS option tells LOADPLUS to pause after the PRELOAD phase only if input records were discarded during the PRELOAD phase. You can restart the utility in the LOAD phase.

**RETCODE**

RETCODE allows you to designate the return code that LOADPLUS assigns when pausing after the PRELOAD phase due to discards encountered in the PRELOAD phase. Although you can specify any positive integer, note the following considerations for certain values:

- LOADPLUS always ends with the highest return code assigned during the load job. Therefore, because PRELOAD PAUSE normally ends with return code 4, specifying a value of 4 or less has no effect. For more information, see “Return code hierarchy in LOADPLUS” on page 158.
If you specify an integer that is greater than 31 but less than 4096, LOADPLUS issues a user abend that is equal to the integer specified.

If you specify an integer that is greater than 4095, the LOADPLUS job step terminates with a system abend 001.

For information about return codes that LOADPLUS normally returns, access the messages section of the BMC Documentation Center from the BMC Support Central site (http://www.bmc.com/support). For information about specifying a user-defined return code for a full load job when discard violations occur, see “IFDISCARDS” on page 223.

**INDSN**

INDSN enables dynamic allocation for your input (SYSREC) data sets.

Specify from 1 through 256 data set names, enclosed in single quotation marks, and separated by commas. LOADPLUS allocates the input data sets for use during the load job.

Use this option for one or more of the following reasons:

- When loading data from UNLOAD PLUS by using FORMAT BMCUNLOAD
- To load data from different input data sets by updating only your syntax cards
- To eliminate the need to override SYSREC DD statements in your cataloged JCL procedures (PROCs) with SYSREC DD statements in your JCL

The input data sets must already exist and must be cataloged tape or DASD data sets.

**Restrictions**

The following restrictions apply to using the INDSN option:

- You cannot use this option when loading data from a batch pipe.
- You cannot specify multiple data sets that are on a stacked tape.
- You cannot specify any parameters for these data sets (such as VOLSER, DCB, or UNIT) other than the data set name.

- If you include SYSREC DD statements in your JCL when you specify this option, LOADPLUS issues message BMC50460E and terminates.

**Additional considerations**

The following additional considerations apply to the INDSN option:

- You can specify a GDG name as your data set name. LOADPLUS interprets a digit, plus sign, or hyphen following an open parenthesis in `dataSetName` as an indication that the data set name is a GDG name.

- You can specify a PDS name as your data set name as long as you include a member name. LOADPLUS interprets an alphabetic character following an open parenthesis in `dataSetName` as the beginning of a PDS member name.

- You can specify a dummy data set by specifying INDSN('NULLFILE').

- You can use the INDDN option to override the ddname prefix for your input data sets during dynamic allocation.

- Because each data set name that you specify with INDSN is assigned to a separate SYSREC data set, all restrictions and considerations that apply to multiple SYSREC data sets apply to specifying multiple data set names on the INDSN option. For more information, see “SYSREC data sets in LOADPLUS” on page 456.

**INDDN**

The INDDN option allows you to override the default ddname (SYSREC) or ddname prefix of the input data set.

```
INDDN  SYSREC  ddbname
```

The value for INDDN tells LOADPLUS which DD statements in your JCL are input data sets. For more information about these data sets, see “SYSREC data sets in LOADPLUS” on page 456.

**Multiple data sets**
If you use multiple input data sets, specify only the ddname prefix (no *nn*) in this option. Note the following considerations:

- If you are using the INDSN option to dynamically allocate input data sets, LOADPLUS appends the data set number to the prefix that you specify. This prefix plus the highest data set number must not exceed eight characters.

- If you are not dynamically allocating input data sets, you must append *nn* to the ddnames in the DD statements in your JCL, where *nn* can be one or two valid alphanumeric or national characters.

- If you are not dynamically allocating input data sets, the prefix must allow LOADPLUS to differentiate the prefix from any ddnames that you specify in your JCL for other data sets. For example, if you specify an INDDN prefix of SYSU and your JCL contains a DD statement for your work data set that uses a ddname of SYSUT1, LOADPLUS assumes that the SYSU prefix refers to the SYSUT1 data set.

### Additional considerations

Note the following additional considerations when you use this option:

- If you also specify INDSN on your LOAD command, LOADPLUS uses the value for INDDN as the ddname or ddname prefix when constructing your input data set DD statements. If any DD statements in your JCL match the INDDN value, LOADPLUS issues message BMC50460E and terminates.

  For a DSNUTILB load, LOADPLUS uses this value as the ddname.

- If you do not specify INDSN on your LOAD command and you use the default ddname SYSREC in your JCL, you do not need to use this option. However, if you want to use a ddname other than SYSREC, you must specify it both in this option and in your JCL.

### Specifying the default

You can specify the default for the INDDN command option in your installation options module by using the INDDN installation option. LOADPLUS was shipped with a default value of SYSREC for this option. The command option overrides the default that is in the installation options module.

### LOADDN

The LOADDN option allows you to override the default ddname (SORTOUT) or ddname prefix for the output data set from the PRELOAD phase.
For more information about these data sets, see “SORTOUT data sets in LOADPLUS” on page 443.

If you use this command option to override the default name in the installation options module, you must also change the ddname in your JCL.

**Multiple data sets**

For multiple SORTOUT data sets, specify only the ddname prefix (no n) in this option. Note the following considerations:

- If you are not dynamically allocating output data sets, you must append n to the ddnames in the DD statements in your JCL, where n is any valid alphanumeric or national character. These ddnames must not exceed eight characters.

- If you are not dynamically allocating output data sets, the prefix must allow LOADPLUS to differentiate it from any ddnames that you specify in your JCL for other data sets. For example, if you specify a LOADDN prefix of SYSU and your JCL contains a DD statement for your work data set that uses a ddname of SYSUT1, LOADPLUS assumes that the SYSU prefix refers to the SYSUT1 data set.

- When you dynamically allocate SORTOUT data sets, LOADPLUS appends the data set number to the ddname prefix that you specify. To dynamically allocate more than nine SORTOUT data sets, use this option to specify a ddname prefix. The prefix plus the highest data set number must not exceed eight characters. For more information, see “Specifying ddname prefixes” on page 81.

- If dynamic data set allocation is active and you specify more than one ddname prefix for dynamic allocation, the prefix for each ddname must be different enough for LOADPLUS to differentiate one prefix from another. To be different enough, if these prefixes are different only because one prefix has additional trailing bytes, then these trailing bytes must contain at least one nonnumeric byte. For example, the first set of prefixes that follow is different enough, but the second set is not:

  — Acceptable set:

  BMCRD
  BMCRDWK
Additional considerations

The following considerations apply to the LOADDN option:

- If you specify or default to a prefix with a length of 7 or more characters, and the value for the SMAX installation option is greater than 9, LOADPLUS reduces the value of SMAX to 9.

- When invoking DSNUTILB, LOADPLUS uses this option only to pass a value for the &DDNAME variable for the data set name pattern. LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

Specifying the default

You can specify the default for the LOADDN command option in your installation options module by using the LOADDN installation option. LOADPLUS was shipped with a default value of SORTOUT for this option. The command option overrides the default that is in the installation options module.

WORKDDN

The WORKDDN option allows you to override the default ddname (SYSUT1) or ddname prefix of the work data set.

[Diagram showing WORKDDN and SYSUT1]

For more information about these data sets, see “SYSUT1 data sets in LOADPLUS” on page 459.

If you use this command option to override the default name in the installation options module, you must also change the ddname in your JCL.

DSNUTILB load jobs

When invoking DSNUTILB, LOADPLUS uses this option only to pass a value for the &DDNAME variable for the data set name pattern. LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.
**Multiple data sets**

If you use multiple work data sets, specify only the ddname prefix (no \( nn \)) in this option. Note the following considerations:

- If you are not dynamically allocating work data sets, you must append \( nn \) to the ddnames in the DD statements in your JCL, where \( nn \) can be one or two valid alphanumeric or national characters. These ddnames must not exceed eight characters.

- If you are not dynamically allocating work data sets, the prefix must allow LOADPLUS to differentiate it from any ddnames that you specify in your JCL for other data sets. For example, if you specify a WORKDDN prefix of SYSR and your JCL contains a DD statement for your input data set that uses a ddname of SYSREC, LOADPLUS assumes that the SYSR prefix refers to the SYSREC data set.

- When you dynamically allocate SYSUT1 data sets, LOADPLUS appends the data set number to the ddname prefix that you specify. To dynamically allocate more than 99 SYSUT1 data sets, use this option to specify a ddname prefix. The prefix plus the highest data set number must not exceed eight characters. For more information, see “Specifying ddname prefixes” on page 81.

- If dynamic data set allocation is active and you specify more than one ddname prefix for dynamic allocation, the prefix for each ddname must be different enough for LOADPLUS to differentiate one prefix from another. To be different enough, if these prefixes are different only because one prefix has additional trailing bytes, then these trailing bytes must contain at least one nonnumeric byte. For example, the first set of prefixes that follow is different enough, but the second set is not:

  — Acceptable set:
  
  - BMCRD
  - BMCRDWK

  — Unacceptable set:
  
  - BMCRD
  - BMCRD11

**Specifying the default**

You can specify the default for the WORKDDN command option in your installation options module by using the WORKDDN installation option. LOADPLUS was shipped with a default value of SYSUT1 for this option. The command option overrides the default that is in the installation options module.

**ERRDDN**

The ERRDDN option allows you to override the default data set ddname (SYSERR).
This data set contains information about records that are discarded because of errors. For detailed information about using the SYSERR data set in LOADPLUS, see “SYSERR data set in LOADPLUS” on page 453.

If you use this command option to override the default name in the installation options module, you must also change the ddname in your JCL.

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS uses this option only to pass a value for the &DDNAME variable for the data set name pattern. LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

**Specifying the default**

You can specify the default for the ERRDDN command option in your installation options module by using the ERRDDN installation option. LOADPLUS was shipped with a default value of SYSERR for this option. The command option overrides the default that is in the installation options module.

**DISCARDDN**

The DISCARDDN option allows you to override the default ddname (SYSDISC) of the discard data set.

For detailed information about using the SYSDISC data set in LOADPLUS, see “SYSDISC data set in LOADPLUS” on page 450.

If you use this command option to override the default name in the installation options module, you must also change the ddname in your JCL.
**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS uses this option only to pass a value for the &DDNAME variable for the data set name pattern. LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

**Specifying the default**

You can specify the default for the DISCARDDDN command option in your installation options module by using the DISCDDDN installation option. LOADPLUS was shipped with a default value of SYSDISC for this option. The command option overrides the default that is in the installation options module.

**DISCARDS**

The DISCARDS option defines the limit on the number of discard violations (see the description of discard violations in this topic).

When LOADPLUS reaches the specified limit, the load job terminates.
The default is 0 (no limit) when you do not specify the DISCARDS keyword. When you specify the DISCARDS keyword, you must supply a value and this value must be an integer (which can be 0).

You can also specify a user-defined return code for LOADPLUS to use when you reach this limit. (See “IFDISCARDLIMIT” on page 224.)

Additional considerations

The following considerations apply to this option:

■ When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

■ Unless you are running an SQLAPPLY load job, this option is not valid for a single-phase load (PRELOAD LOAD) when you specify LOAD RESUME YES.

■ For a two-phase load, a value greater than 0 for this option is meaningful only with one of the following options:
  — UNIQUECHECK YES (which is the default)
  — UNIQUECHECK CLUSTER when all participating unique indexes are data-sorting indexes

For information, see “UNIQUECHECK” on page 244.

Discard violations

Discard violations are calculated as the total number of violations that cause LOADPLUS to discard input records during the PRELOAD or COMBINED phase for a particular load job. For example, if two input records are duplicates and they both violate a check constraint, LOADPLUS reports four discard violations (assuming that you did not specify IGNORE DUPKEY or IGNORE CHKC). In this case, however, the SYSDISC data set contains only the two discarded input records.

LOADPLUS generates discard violations for any of the following reasons:

■ An input value position is past the end of the input record.

■ An input value is not entirely contained on the input record.

■ A data conversion error (including a FIELDPROC error) occurs.

■ A value generated for an identity column is out of range.

■ A verification record does not match the table that is being loaded (FORMAT BMCUNLOAD).
- LOADPLUS encounters an error byte during data translation.
- LOADPLUS encounters a substitution character during data translation, but you specified NOSUBS.
- An expansion error occurs during data translation.
- A VALIDPROC violation occurs.
- A record does not match any WHEN specification or table OBID.
- A record does not match any partition that is being loaded.
- A DB2 check constraint violation occurs.
- A DB2 referential constraint violation occurs.
- An informational referential constraint violation occurs (and INFORI is YES).
- A unique key violation occurs in the PRELOAD or COMBINED phase.
- An SQLCODE -330, -530, or -803 occurs during an SQLAPPLY load.
- A problem occurs with a referenced file that is used to load LOB or XML data.
- A problem occurs with an XML document.
- A conversion error occurs on a key that is derived from an expression.

Records in the cases described in the following table are discarded but not recorded as discard violations. The discarded records are not written to the discard data set, and they do not count toward the DISCARDS limit.

### Table 38: Discarded records that are not written to SYSDISC

<table>
<thead>
<tr>
<th>Condition</th>
<th>Type of discarded records not written to SYSDISC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input data set is a pipe</td>
<td>Records that meet any of the listed discard criteria</td>
</tr>
<tr>
<td>UNIQUECHECK NO specified for a two-phase load job</td>
<td>Duplicate records</td>
</tr>
<tr>
<td>UNIQUECHECK CLUSTER specified for a two-phase load job</td>
<td>Duplicates in non-data-sorting indexes</td>
</tr>
<tr>
<td>LOADPLUS evaluates the same input file for multiple INTO statements</td>
<td>Input records that are valid for at least one INTO statement, but not all INTO statements</td>
</tr>
</tbody>
</table>
### IGNORE

The IGNORE option tells LOADPLUS to ignore the specified discard types during the PRELOAD phase or the COMBINED phase. If the specified type of discard occurs, LOADPLUS

- Does not write an error record to the SYSERR data set (unless the discard is a duplicate key error or referential constraint violation)
- Does not report the error in the error summary report (unless the discard is a referential constraint violation)
- Does not write the corresponding input record to the discard data set (unless the record also violates another discard type)

An ignored discard violation does not count toward the DISCARDS limit.

**Note**

If you use MSGLEVEL(1), LOADPLUS issues messages BMC51501E, BMC51502E, BMC51503E, BMC51505E, and BMC51506E to help diagnose problems regardless of the IGNORE option.

The IGNORE option is especially useful for WHEN and PART discards when all records in your input file do not participate in the load.

The following table lists the keywords and discard types, along with their descriptions, that you can specify with the IGNORE option:

<table>
<thead>
<tr>
<th>IGNORE keyword</th>
<th>Types of records for which discards are ignored</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
<td>Any discards</td>
</tr>
<tr>
<td>WHEN</td>
<td>Records that are not selected by any INTO statement or that are discarded as a result of a table or column definition mismatch when specifying FORMAT UNLOAD, FORMAT BMC, or FORMAT BMCUNLOAD</td>
</tr>
<tr>
<td>PART</td>
<td>Records that are not selected by any partitions being loaded</td>
</tr>
<tr>
<td>FIELDSPEC</td>
<td>Records that are discarded due to an error on the field specification</td>
</tr>
<tr>
<td>CONV</td>
<td>Records that are discarded due to an error in conversion</td>
</tr>
<tr>
<td>DUPKEY</td>
<td>Records that are discarded due to duplicate keys</td>
</tr>
<tr>
<td>VALPROC</td>
<td>Records that are discarded because of a validation procedure</td>
</tr>
<tr>
<td>CHKKC</td>
<td>Records that are discarded because of check constraints</td>
</tr>
<tr>
<td>FRGNKEY</td>
<td>Records that are discarded because of an invalid value in a foreign key</td>
</tr>
</tbody>
</table>
Restriction

This option is not valid when LOADPLUS invokes DSNUTILB. LOADPLUS issues message BMC50178E and terminates.

REPORT

The REPORT option tells LOADPLUS which discard types to report or not report in the error summary report. Although this option has no effect on actual discard processing, it allows you to limit the number of SYSPRINT records when you expect many discards.

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.

YES

When you specify DISCARDS REPORT YES, the discard types that you specify are reported in the LOADPLUS error summary report. DISCARDS REPORT YES ALL is the default.

LOADPLUS does not report any discard types that you specify with the IGNORE option, even if you specify them in the REPORT YES option.

NO

The discard types that you specify in this option are not reported in the LOADPLUS error summary report. This option is beneficial when you expect many discards.

REPORT NO has no effect on actual discard processing. Depending on the use of the IGNORE option, LOADPLUS still writes error records to the SYSERR data set and still writes corresponding input records to the discard data set.

discardType

The following table lists the keywords and discard types, along with their descriptions, that you can specify with the REPORT option:

<table>
<thead>
<tr>
<th>IGNORE keyword</th>
<th>Types of records for which discards are ignored</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDERROR</td>
<td>Records that are outside the range that is defined on the identity column</td>
</tr>
</tbody>
</table>
Table 40: Valid discard types used with the REPORT option

<table>
<thead>
<tr>
<th>REPORT keyword</th>
<th>Types of records for which discards are not reported</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
<td>All discards</td>
</tr>
<tr>
<td>WHEN</td>
<td>Records that are not selected by any INTO statement or that are discarded as a result of a table or column definition mismatch when specifying FORMAT UNLOAD, FORMAT BMC, or FORMAT BMCUNLOAD</td>
</tr>
<tr>
<td>PART</td>
<td>Records that are not selected by any partitions being loaded</td>
</tr>
<tr>
<td>FIELDSPEC</td>
<td>Records that are discarded due to an error on the field specification</td>
</tr>
<tr>
<td>CONV</td>
<td>Records that are discarded due to an error in conversion</td>
</tr>
<tr>
<td>DUPKEY</td>
<td>Records that are discarded due to a duplicate key</td>
</tr>
<tr>
<td>VALPROC</td>
<td>Records that are discarded because of a validation procedure</td>
</tr>
<tr>
<td>CHKC</td>
<td>Records that are discarded because of check constraints</td>
</tr>
<tr>
<td>FRGNKEY</td>
<td>Records that are discarded due to an invalid value in a foreign key</td>
</tr>
<tr>
<td>IDERROR</td>
<td>Records that are outside the range that is defined on the identity column</td>
</tr>
</tbody>
</table>

**IFDISCARDS**

The IFDISCARDS option allows you to specify a user-defined return code that LOADPLUS uses upon successful completion of the load job when LOADPLUS discards one or more input records.

You can specify any integer from 0 through 7.

LOADPLUS always ends with the highest return code assigned during the load job. For more information, see “Return code hierarchy in LOADPLUS” on page 158. For information about return codes that LOADPLUS normally returns, access the messages section of the BMC Documentation Center from the BMC Support Central site (http://www.bmc.com/support).

**Restriction**

The following restrictions apply to the IFDISCARDS option:

- When invoking DSNUTILB, LOADPLUS ignores this option.
If you also specify PRELOAD PAUSE, the IFDISCARDS value is not in effect for the paused job. If you restart the job and there are discard violations, LOADPLUS considers the IFDISCARDS value when determining the final return code. For information about specifying a user-defined return code for a PRELOAD PAUSE load job when there are discard violations, see “PAUSE” on page 209.

**Specifying the default**

You can specify the default for the IFDISCARDS command option in your installation options module by using the DISCARDRC installation option. LOADPLUS was shipped with a default value of 0 for this option. The command option overrides the default that is in the installation options module.

### IFDISCARDLIMIT

The IFDISCARDLIMIT option allows you to specify a user-defined return code that LOADPLUS uses when the number of discard violations reaches the discard limit that you specify with the DISCARDS option.

You can specify any integer greater than 7.

LOADPLUS always ends with the highest return code assigned during the load job. For more information, see “Return code hierarchy in LOADPLUS” on page 158. For information about return codes that LOADPLUS normally returns, access the messages section of the BMC Documentation Center from the BMC Support Central site (http://www.bmc.com/support).

**Restriction**

When invoking DSNUTILB, LOADPLUS ignores this option.

**Specifying the default**

You can specify the default for the IFDISCARDLIMIT command option in your installation options module by using the DISCARDLIMRC installation option. LOADPLUS was shipped with a default value of 8 for this option. The command option overrides the default that is in the installation options module.
IFZEROROWS

The IFZEROROWS option allows you to specify a user-defined return code that LOADPLUS uses upon successful completion of the load job when LOADPLUS does not load any rows.

You can specify any integer from 0 through 7.

LOADPLUS always ends with the highest return code assigned during the load job. For more information, see “Return code hierarchy in LOADPLUS” on page 158. For information about return codes that LOADPLUS normally returns, access the messages section of the BMC Documentation Center from the BMC Support Central site (http://www.bmc.com/support).

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.

Specifying the default

You can specify the default for the IFZEROROWS command option in your installation options module by using the ZEROROWRC installation option. LOADPLUS was shipped with a default value of 0 for this option. The command option overrides the default that is in the installation options module.

ORDER

The ORDER option tells LOADPLUS how and whether to order the table rows.
ORDER affects only new rows that you are loading. For a LOAD RESUME YES job, LOADPLUS does not mix new rows with existing rows.

You can always omit the ORDER option from your LOAD command without error. If you do not specify ORDER, the default is ORDER NO except when you are loading a multi-table, segmented table space. In this case, ORDER YES is the default.

**Restrictions**

The following restrictions apply to the ORDER option:

- When invoking DSNUTILB, LOADPLUS ignores this option.
- You cannot change the value of the ORDER option when you restart your load job.

**NO**

If you specify ORDER NO, LOADPLUS performs no ordering at all, and the rows retain the order of the input data set from which they were read. The table space might require reorganization for adequate DB2 performance. LOADPLUS sorts all indexes independently of the data.

ORDER NO is the default except when you are loading a multi-table, segmented table space. (In this case, ORDER YES is the default.)

**Restrictions**

In addition to the restrictions for the ORDER option, the following restrictions apply specifically to ORDER NO:
Do not specify this option if you are loading multiple tables in a segmented table space unless you are performing an SQLAPPLY load. You can specify this option if you are loading one table of a multi-table, segmented table space.

For optimal performance, BMC recommends that you do not specify this option if a large number of partitions are participating in the load job.

**PRESORTED**

If you specify ORDER PRESORTED, LOADPLUS verifies that the rows are in the correct clustering index order. If the data is partitioned and the clustering index is either not partitioned or is a data-partitioned secondary index (DPSI), the data must be sorted first by partition, then by clustering key.

LOADPLUS performs no sorts on either the data or the index and the job terminates if the data is not in the correct order.

The encoding scheme of the table determines the collating sequence for character data. For example, if the table is defined as EBCDIC, the collating sequence will be EBCDIC.

ORDER PRESORTED is valid when loading a single table with one of the following specifications:

- LOAD REPLACE
- LOAD RESUME NO
- LOAD RESUME YES INTO TABLE `tableName` REPLACE
- Single-phase LOAD RESUME YES PART `n` REPLACE if you are replacing all partitions that are participating in the load and no nonpartitioned indexes are participating in the load

**Restrictions**

In addition to the restrictions for the ORDER option, the following restrictions apply specifically to ORDER PRESORTED:

- If you use this option with types of load jobs other than the ones described as valid for ORDER PRESORTED, LOADPLUS issues message BMC50115E or BMC51430E and terminates.
- You must specify only one SYSREC data set when using ORDER PRESORTED. Otherwise, the job terminates.
YES

If you specify ORDER YES, LOADPLUS orders the rows in the tables of the table space by their clustering key or, if there is no clustering key, by partitioning key.

Note

To simplify terminology, this book refers to an index that is used to sort data as a data-sorting index. For traditional table spaces, this is a clustering index. For table-controlled partitioned table spaces, this is either a clustering index or partitioning index.

The encoding scheme of the table determines the collating sequence for character data. For example, if the table is defined as EBCDIC, the collating sequence will be EBCDIC.

Restriction

In addition to the restrictions for the ORDER option, if you specify ORDER YES and also specify SKIPIX to skip a clustering index, LOADPLUS terminates with message BMC51527E.

ASSOCIATE

Use the ASSOCIATE option to specify how you want LOADPLUS to sort the rows for multi-table table spaces. The following table describes the values that you can specify for the ASSOCIATE option:

Table 41: Values for the ASSOCIATE option

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>BYTABLE</td>
<td>BYTABLE, which is the default if you specify ORDER YES, tells LOADPLUS to sort the rows by table as well as by each table’s clustering key. If no clustering key exists, LOADPLUS uses X’00’s. LOADPLUS always sorts segmented table spaces by table.</td>
<td>This option is meaningless for a single-table table space with no clustering index. In this case, LOADPLUS processes this option as if you specified ORDER NO</td>
</tr>
</tbody>
</table>
## ENUMROWS

The ENUMROWS option provides estimated information that LOADPLUS and BMCSORT use to dynamically allocate your data sets and to determine the most efficient method for performing sort processing.

If the value for the AUTOENUMROWS installation option is NO, specify the ENUMROWS command option for any of the following scenarios:

- You specify the ANALYZE option.
  
  You must specify either AUTO or the appropriate `total` or `new` parameter so that LOADPLUS can accurately estimate the size of the work files.

- Dynamic data set allocation is active.
  
  You must specify either AUTO or the appropriate `total` or `new` parameter so that LOADPLUS can accurately estimate the size of the work files.
- BMCSORT is dynamically allocating your sort work data sets.
  
  BMC recommends that you specify ENUMROWS, with either the AUTO option or the appropriate total or new parameter, to allow BMCSORT to most accurately and efficiently allocate your sort work data sets.

  **Note**

  When BMCSORT is dynamically allocating your sort work data sets, BMC recommends that you also specify a value greater than 0 for the SORTNUM installation or command option.

- You are loading a very large number of rows.
  
  In this case, BMC strongly recommends that you specify ENUMROWS, with either the AUTO option or the appropriate total or new parameter, so that BMCSORT can determine the most efficient method for performing sort processing.

**Specifying total, new, or error values**

BMC recommends that you enable automated file size estimation by specifying either AUTO for the ENUMROWS command option or YES for the AUTOENUMROWS installation option. However, if you choose to specify your own estimates, consider the information in this section.

Use the following table to determine whether to specify a total or new value:

**Table 42: Whether to specify total or new with ENUMROWS**

<table>
<thead>
<tr>
<th>LOAD RESUME or REPLACE</th>
<th>total or new</th>
<th>Additional information</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD REPLACE</td>
<td>Either total or new</td>
<td>If you specify a value of 0 for both parameters, LOADPLUS assumes that you are loading zero rows. However, if you specify 0 for only one parameter, LOADPLUS uses the value of the other parameter. If you specify a value for both parameters, LOADPLUS uses the largest value.</td>
</tr>
<tr>
<td>LOAD RESUME YES (including LOAD RESUME YES PART n REPLACE or LOAD RESUME YES INTO TABLE tableName REPLACE)</td>
<td>new</td>
<td>If you specify a value of 0 for new, LOADPLUS assumes that you are loading zero rows unless you also specify a value for the total parameter. If you also specify a value for the total parameter, LOADPLUS calculates a value for the new parameter based on the total parameter and the number of existing rows.</td>
</tr>
</tbody>
</table>
Note the following additional information for specifying `total`, `new`, or `error` values:

- If you specify values for the `total`, `new`, or `error` parameters, note that they are positional. The following example illustrates specifying only the `new` parameter:

  ```
  ENUMROWS (,15000)
  ```

- Do not include commas in the numbers that you specify.

- For any of the ENUMROWS values, BMC recommends that you overestimate rather than underestimate. Overestimating results in LOADPLUS allocating files that are larger than needed, but underestimating can cause your job to terminate.

- For partitioned table spaces, LOADPLUS assumes an even distribution of rows across the partitions being loaded and divides the number that you supply by the number of partitions. If rows are not evenly distributed across partitions, consider using the following formula to determine the value to supply for ENUMROWS:

  ```
  est rows for largest partition * number of partitions to be loaded
  ```

**Restriction**

When invoking DSNUTILB, LOADPLUS ignores this option.

**Specifying the default**

You can specify the default behavior for automated file size estimation in your installation options module by using the AUTOENUMROWS installation option. LOADPLUS was shipped with a default value of YES for this option. Any value that you specify for the ENUMROWS command option overrides the AUTOENUMROWS value that is in the installation options module.

**AUTO**

This option tells LOADPLUS to automatically estimate the size of the input files whose ddnames you supply in your JCL. If you are loading multiple input files, LOADPLUS issues a separate message BMC51561I for each input file, indicating the calculated number of input records for that file.

**Additional considerations**

Note the following information about automated file size estimation:

- For files with variable-length records, LOADPLUS estimates data in variable-length columns as 50 percent of the defined column length. If you know that this will result in an over- or under-allocation, use the `percent` parameter to adjust the file size estimation.

- LOADPLUS supports all major tape management systems. However, if you use the BMC Control-M/Tape product as your tape management system, the Control-M/Tape product imposes additional restrictions on your tape load job. For example, LOADPLUS assumes that the load job is running in real time, which might not always be true for larger loads.
M/Tape load library must be in your system LINKLIST or your JOBLIB or STEPLIB libraries. If this load library is not available, you might receive inaccurate file size estimations for tape input data sets.

- When any of the following conditions exists, LOADPLUS might not accurately estimate your input file size. In these cases, LOADPLUS issues message BMC51562W with an explanation of the condition. LOADPLUS continues processing, but might not provide optimal allocation of your work data sets or might not allocate sort processes efficiently.

---

- The input file is a member of a PDS or PDSE.
  LOADPLUS cannot determine the file size of a PDS or PDSE member. In this case, LOADPLUS estimates a standard file size of 40 MB per file. If this size is not a reasonable value, use the *percent* parameter to adjust the estimate.

- The files span more than five volumes.
  For volumes after the fifth volume, LOADPLUS estimates the file size as the average of the first five volumes.

- The input data is compressed.
  LOADPLUS attempts to adjust its estimate for compression, but cannot determine the compression ratio. If you understand how compression affects your data, use the *percent* parameter to adjust the calculation.

- The input file is a pipe data set.
  LOADPLUS cannot determine the file size of a pipe data set. In this case, LOADPLUS estimates a file size of 0. If you are performing a load job that needs an ENUMROWS specification (such as dynamically allocating work files), and your input file is a pipe data set, specify a numeric value for ENUMROWS.

- The input file is empty or is a dummy data set.
  If your input file is empty or you specify DD DUMMY, LOADPLUS cannot correctly determine the record format or file size. In this case, BMC recommends that you specify ENUMROWS 0 instead of AUTO.

- If you are using UNLOAD PLUS with FORMAT BMCLOAD and LOADPLUS with FORMAT BMCUNLOAD, consider specifying ENUMROWS as a string on the UNLOAD PLUS CNTLCARDS option instead of automating file size estimation. In this case, UNLOAD PLUS provides the number of rows that it has unloaded with the LOADPLUS ENUMROWS option that it generates. For an example, see “Example 14 LOAD REPLACE from UNLOAD PLUS data in internal format” on page 588.

- When estimating file size, LOADPLUS does not consider the following items:
  - Discarded data, including discards due to WHEN processing
— Nonparticipating partitions
To adjust for these conditions, use the percent parameter.

percent

This parameter allows you to specify the percentage of the calculated file size to use as an estimate. You can specify a value from 1 through $2^{31}$ and you do not need to specify a percent sign. The default value is 100.

Use this parameter to fine-tune automated file size estimation. For example, you might want to use this parameter in the following situations:

- Under conditions that LOADPLUS does not take into account, such as discarded data
- When specifying multiple INTO statements for different tables
  In this case, LOADPLUS estimates file size based on an assumption that each input record will be loaded into each table. If you know that each input record will be loaded into only one table (for example, if you specify UNIQUEINTO), you can use this parameter to reduce the estimated file size.
- When specifying multiple INTO statements for the same table
  LOADPLUS estimates file size by multiplying the estimated number of input records by the number of tables being loaded. Because only one table is being loaded, LOADPLUS might underestimate the file size in this case. Use this parameter to increase the estimated file size.
- When the difference between the calculated number of input records reported by message BMC51561I and the actual number of records reported by message BMC51478I is significant (for example, when loading variable-length files)
  In this case, use the following formula to determine a value to specify:

  \[
  \text{percent} = \frac{\text{actual} \times 100}{\text{calculated}}
  \]

  total

This value is an integer that specifies the total number of rows that you expect to have in all tables of the table space following the load. A value of 0 tells LOADPLUS to dynamically allocate data sets of a minimal size for those DDTYPEs for which dynamic allocation is active.

If BMCSORT is dynamically allocating sort work data sets, LOADPLUS passes this integer to BMCSORT through the SIZE sort parameter.
**new**

This value is an integer that specifies the number of records in your SYSREC data set. A value of 0 tells LOADPLUS to dynamically allocate data sets of a minimal size for those DDTYPEs for which dynamic allocation is active.

**Additional considerations**

Note the following additional considerations:

- For multi-table table spaces, LOADPLUS multiplies the value that you specify by the number of tables that you are loading.
- For LOAD RESUME YES PART \( n \) REPLACE INDEX UPDATE, \( new \) is the total number of rows that you expect to have in the partitions that you are replacing.

**error**

This value is an integer that specifies the number of rows that you expect to be discarded. The default is 32000 or the number of rows being loaded if that number is less than 32000. Specify a value for error only if you expect the number of discards to be greater than 32000. This option applies only if LOADPLUS is dynamically allocating the SYSERR or SYSDISC data sets.

---

**Note**

If you specify the DISCARDS option with a value greater than 0 and you are dynamically allocating SYSDISC, LOADPLUS uses the DISCARDS value instead of the value for error to determine the size of the SYSDISC data set.

---

**SORTDEVT**

The SORTDEVT option specifies the device type for the sort work files that are allocated dynamically.

```
SORTDEVT deviceType
```

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

For non-DSNUTILB load jobs, if the first parameter of the SORTDEVT installation option is null, this command option overrides the first parameter of the BMCSORT DYNALOC installation option. If the value of the third parameter in the BMCSORT DYNALOC installation option is OFF, specifying a value for the SORTDEVT option turns BMCSORT dynamic allocation on.
Specifying the default

You can specify the default for the SORTDEVVT command option in your installation options module by using the SORTDEVVT installation option. LOADPLUS was shipped with a default value of (, SYSALLDA) for this option (where the first parameter affects non-DSNUTILB jobs and the second parameter affects DSNUTILB jobs). The command option overrides the default that is in the installation options module for both parameters.

SORTNUM

The SORTNUM option affects the allocation of sort work files in the following cases.

You can specify any integer value from 0 through 255.

DSNUTILB load jobs

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility as the number of sort work files to allocate dynamically. For this type of load, the value must be 2 or greater.

All other load jobs

This value is in effect when BMCSORT is allocating your sort work files dynamically.

The following table describes the action that BMCSORT takes for each value that you can specify for this option. The table also provides any additional considerations for these values.

Table 43: SORTNUM values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Additional considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>BMCSORT honors the value of the third parameter of the BMCSORT DYNALOC installation option. (This parameter tells BMCSORT whether to dynamically allocate sort work files.)</td>
<td>For more information about how this parameter affects dynamic allocation, see “SORTWK data sets in LOADPLUS” on page 446.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
<td>Additional considerations</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1-32</td>
<td>BMCSORT dynamically allocates the number of sort work files that it needs, up to 32 minus any preallocated sort work files. This number is per sort task.</td>
<td>Preallocated sort work files include sort work files that are allocated in your JCL and any sort work files that LOADPLUS dynamically allocates.</td>
</tr>
<tr>
<td>33-255</td>
<td>BMCSORT dynamically allocates the number of sort work files that it needs, up to the number that you specify minus any preallocated sort work files. This number is per sort task.</td>
<td></td>
</tr>
</tbody>
</table>

If the value of the third parameter in the BMCSORT DYNALOC installation option is OFF, specifying a value greater than 0 for the SORTNUM option turns BMCSORT dynamic allocation on, and BMCSORT allocates sort work files as needed.

For information about when BMCSORT allocates your sort work files dynamically, see “SORTWK data sets in LOADPLUS” on page 446.

**Specifying the default**

You can specify the default for the SORTNUM command option in your installation options module by using the SORTNUM installation option. LOADPLUS was shipped with a value of 32. The command option overrides the default that is in the installation options module.

**MAXSORTS**

The MAXSORTS option specifies the maximum number of each type of task that LOADPLUS can run concurrently.

```
MAXSORTS 16
```

MAXSORTS controls only the tasks shown in the following table:

<table>
<thead>
<tr>
<th>Phase</th>
<th>Task types</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANALYZE</td>
<td>Any</td>
</tr>
</tbody>
</table>
Restrictions

Note the following restrictions on the MAXSORTS option:

- MAXSORTS does not override the value of the SMAX installation option except in the following case: if you specify a MAXSORTS value that is smaller than the SMAX value, MAXSORTS overrides SMAX for the types of tasks that MAXSORTS controls.

- In either of the following cases, LOADPLUS ignores the MAXSORTS option:
  - You are running an SQLAPPLY load.
  - LOADPLUS is invoking DSNUTILB.

**AVAILPAGEPCT**

LOADPLUS uses the AVAILPAGEPCT option to control virtual storage that it allocates to BMCSORT for concurrent sort processing.

AVAILPAGEPCT specifies the maximum percentage of available 4-KB pages, as obtained from the system, that LOADPLUS can allocate. You can specify any integer from 0 through 100.

**Note**

LOADPLUS defines *available pages* as pages that have not been used. *Total pages* (which you can control with the TOTALPAGEPCT option), are pages that are underutilized and are available for use.

A value of 0 tells LOADPLUS to ignore the number of available pages when allocating sort memory. A value of 1 through 100 tells LOADPLUS to use up to the specified percentage of available pages when allocating sort memory. For example, AVAILPAGEPCT 50 tells LOADPLUS to use no more than 50 percent of the available pages.
Additional considerations

The following additional information applies to the AVAILPAGEPCT option:

- Because available pages are rarely subject to system paging, changing this value will have a minimal effect, if any, on system performance.

- When you specify values greater than 0 for both AVAILPAGEPCT and TOTALPAGEPCT, LOADPLUS uses the lesser of the two calculated results as the maximum amount of memory for sort processing.

- If LOADPLUS cannot perform an optimal sort due to an insufficient number of available or total pages that it is enabled to allocate, the SHORTMEMORY option controls the action that LOADPLUS takes.

- When invoking DSNUTILB, LOADPLUS ignores this option.

Specifying the default

You can specify the default for the AVAILPAGEPCT command option in your installation options module by using the AVAILPAGEPCT installation option. LOADPLUS was shipped with a default value of 0 for this option. The command option overrides the default that is in the installation options module.

TOTALPAGEPCT

LOADPLUS uses the TOTALPAGEPCT option to control virtual storage allocated to BMCSORT for concurrent sort processing.

TOTALPAGEPCT specifies the maximum percentage of total 4-KB pages, as obtained from the system, that LOADPLUS can allocate. You can specify any integer from 0 through 100.

Note

LOADPLUS defines total pages as pages that are underutilized and are available for use. Available pages (which you can control with the AVAILPAGEPCT option), are pages that have not been used.
A value of 0 tells LOADPLUS to ignore the number of total pages when allocating sort memory.

A value of 1 through 100 tells LOADPLUS to use up to the specified percentage of total pages when allocating sort memory. For example, TOTALPAGEPCT 50 tells LOADPLUS to use no more than 50 percent of the total pages.

**Additional considerations**

The following additional information applies to the TOTALPAGEPCT option:

- Because total pages are subject to more system paging than available pages, changing the value for this option is more likely to affect system performance than changing the value for the AVAILPAGEPCT option.

- When you specify values greater than 0 for both TOTALPAGEPCT and AVAILPAGEPCT, LOADPLUS uses the lesser of the two calculated results as the maximum amount of memory for sort processing.

- If LOADPLUS is unable to perform an optimal sort due to an insufficient number of available or total pages that it is enabled to allocate, the SHORTMEMORY installation or command option controls the action that LOADPLUS takes.

- When invoking DSNUTILB, LOADPLUS ignores this option.

**Specifying the default**

You can specify the default for the TOTALPAGEPCT command option in your installation options module by using the TOTALPAGEPCT installation option. LOADPLUS was shipped with a default value of 30 for this option. The command option overrides the default that is in the installation options module.

**SHORTMEMORY**

The SHORTMEMORY option controls the action that LOADPLUS takes when certain memory shortages exist.

LOADPLUS takes the specified action when one of the following memory shortages exists during sort processing:

- The system contains insufficient total or available pages of memory for LOADPLUS to perform an optimal sort.
- The region contains insufficient memory for LOADPLUS to perform a minimum number of tasks, given the amount of memory required to perform an optimal sort. However, the region does contain at least 1280 KB of memory, or the minimum amount of memory specified by the MINSORTMEMORY option. If the region contains 1280 KB of memory and the amount of memory specified by MINSORTMEMORY is available, LOADPLUS uses the greater amount of memory.

The following table shows the relationship between SHORTMEMORY values and the following conditions:

- Memory data obtained from the system
- Memory in the region
- Value specified for the SMCORE installation option
- Value specified for the MINSORTMEMORY installation or command option

Table 45: Action LOADPLUS takes when memory resources are constrained during sort processing

<table>
<thead>
<tr>
<th>Location of memory</th>
<th>Amount of memory is</th>
<th>SHORTMEMORY CONTINUE</th>
<th>SHORTMEMORY FAIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory in the system</td>
<td>Insufficient to run a minimum number of optimal sort tasks based on the amount of data to be sorted</td>
<td>LOADPLUS runs a minimum number of tasks with 1280 KB of memory or the amount of memory that you specified with MINSORTMEMORY, whichever is greater.</td>
<td>LOADPLUS fails.</td>
</tr>
<tr>
<td></td>
<td>Insufficient as specified by MINSORTMEMORY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Virtual memory in the region</td>
<td>Insufficient to run a minimum number of optimal sort tasks based on the amount of data to be sorted but sufficient as specified by MINSORTMEMORY or at least 1280 KB, whichever is greater</td>
<td>LOADPLUS runs a minimum number of tasks with the available memory.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Insufficient as specified by MINSORTMEMORY or less than 1280 KB, whichever is greater</td>
<td></td>
<td>LOADPLUS fails.</td>
</tr>
<tr>
<td></td>
<td>Insufficient as specified by the first parameter of SMCORE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Restriction
When invoking DSNUTILB, LOADPLUS ignores this option.

**Specifying the default**

You can specify the default for the SHORTMEMORY command option in your installation options module by using the SHORTMEMORY installation option. LOADPLUS was shipped with a default value of CONTINUE for this option. The command option overrides the default that is in the installation options module.

**CONTINUE**

CONTINUE indicates that, when a memory shortage exists, LOADPLUS should issue message BMC50364I, BMC50480I, or BMC50480W and continue sort processing, as described in Table 45 on page 240.

**FAIL**

FAIL indicates that LOADPLUS should fail when a memory shortage exists, issuing message BMC50480E or BMC50399E. Be aware when specifying FAIL that sufficient memory might exist to sort during the PRELOAD phase of a two-phase load. However, because of other system conditions, insufficient available pages might exist during the index build process, which occurs during the LOAD phase.

**MINSORTMEMORY**

The value, in kilobytes, that you specify for the MINSORTMEMORY option helps LOADPLUS determine the minimum amount of memory to allocate to each sort task.

```
+----+----+
| MINSORTMEMORY | 0 |
| integer       |
```

A value of 0 tells LOADPLUS to automatically compute the minimum amount of memory that is needed to optimally perform each sort task. In addition to 0, you can specify any number of kilobytes between 1280 and the value that you specify for the MAXSORTMEMORY installation or command option.

---

**Note**

BMC strongly recommends that you use a value of 0.
For information about how this option interacts with the SMCORE installation option, see “SMCORE=(0K,0K)” on page 749.

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.

Specifying the default

You can specify the default for the MINSORTMEMORY command option in your installation options module by using the MINSORTMEMORY installation option. LOADPLUS was shipped with a default value of 0 for this option. The command option overrides the default that is in the installation options module.

MAXSORTMEMORY

The value, in kilobytes, that you specify for the MAXSORTMEMORY option helps LOADPLUS determine the maximum amount of memory to allocate to each sort task.

A value of 0 tells LOADPLUS to automatically compute the maximum amount of memory that is needed to optimally perform each sort task. In addition to 0, you can specify any number of kilobytes between the value that you specify for the MINSORTMEMORY installation or command option and 2097152.

Note

BMC strongly recommends that you use a value of 0.

For information about how this option interacts with the SMCORE installation option, see “SMCORE=(0K,0K)” on page 749.

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.

Specifying the default

You can specify the default for the MAXSORTMEMORY command option in your installation options module by using the MAXSORTMEMORY installation option.
LOADPLUS was shipped with a default value of 0 for this option. The command option overrides the default that is in the installation options module.

**CONTINUEIF**

The CON TINUEIF option tells LOADPLUS whether the current logical input record continues with the next physical input record.

The two numbers, `start:end`, indicate the starting and ending columns of the continuation indicator in the physical input record. If the comparison of the continuation indicator with the string is equal, LOADPLUS concatenates the next physical input record. You can concatenate any number of physical records into a logical record as long as the length of the logical record does not exceed 32 KB.

You can specify either a character string or a hexadecimal string. The length of the continuation indicator cannot be greater than 70 or less than the length of the string. If you do not specify `:end`, LOADPLUS uses the length of the string. When determining the value to specify, note that in the physical input record, the first column of the record is column 1.

**Restriction**

If you also specify any of the FORMAT options, LOADPLUS issues message BMC51412E and terminates.

**Additional considerations**

The following considerations apply to the CON TINUEIF option:

- The continuation indicator is not included in the logical record. Thus, if you specify CONTINUEIF(73:80)=`string`, the first byte of the first physical record is column 1 of the logical record. The first byte of the second physical record is column 73 of the logical record.

- When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

- When you are loading Unicode data, LOADPLUS assumes that the CON TINUEIF constant that you specify is in EBCDIC format. LOADPLUS translates the constant
to the CCSID of the input data. If you specify a hexadecimal value, it must be in the internal format of the input.

**UNIQUECHECK**

LOADPLUS always checks unique index keys in the input records for duplicate values. The UNIQUECHECK option determines the phase in which LOADPLUS checks for duplicate values and how to handle discarding duplicates.

The following table summarizes the action that LOADPLUS takes based on the value for the UNIQUECHECK option and the type of indexes that are participating in the load job:

**Table 46: UNIQUECHECK options**

<table>
<thead>
<tr>
<th>UNIQUECHECK option</th>
<th>Unique index type</th>
<th>LOADPLUS action on unique key violation</th>
</tr>
</thead>
</table>
| YES                | Any                    | ■ Detects duplicates in the PRELOAD phase  
|                    |                        | ■ Discards incoming records that violate unique key requirement  
|                    |                        | ■ Places records in the discard data set (SYSDISC)                                                     |
| CLUSTER            | Data-sorting           | See the UNIQUECHECK YES option                                                                       |
|                    | Non-data-sorting       | See the UNIQUECHECK NO option                                                                       |
| NO                 | Any                    | ■ Detects duplicates in the LOAD phase  
|                    |                        | ■ Deletes duplicates from the loaded table space and participating indexes, but does not place them in the discard data set (SYSDISC) |

**Restrictions**

The following restrictions apply to the UNIQUECHECK option:
You must specify UNIQUECHECK NO when both of the following conditions exist:
— You specify INDEX UPDATE.
— You are running a two-phase load (PRELOAD CONTINUE).
If you specify UNIQUECHECK YES or CLUSTER, LOADPLUS issues message BMC51419E and terminates.

LOADPLUS ignores this option when any of the following conditions exist:
— You are running an SQLAPPLY load.
  DB2 checks for duplicates during the apply process.
— LOADPLUS invokes DSNUTILB.
  The IBM DB2 LOAD utility checks for duplicates.
— You are running a single-phase load (PRELOAD LOAD)
  In a single-phase load job, LOADPLUS always checks all participating unique indexes for duplicates in the combined phase.

YES

UNIQUECHECK YES (the default) tells LOADPLUS to check for duplicates in key values for all participating unique indexes during the PRELOAD phase. When you specify or default to UNIQUECHECK YES and LOADPLUS detects a unique key violation, LOADPLUS discards the incoming records that violate the unique key requirement and places them in the discard data set (SYSDISC).

CLUSTER

UNIQUECHECK CLUSTER tells LOADPLUS to check for duplicates in key values for only the data-sorting indexes during the PRELOAD phase, and to check for duplicates in non-data-sorting indexes during the LOAD phase. Specifying this option eliminates the need to sort and check other participating indexes in the PRELOAD phase, and can reduce the elapsed time of your load job. When you specify UNIQUECHECK CLUSTER and LOADPLUS detects a unique key violation, the actions that LOADPLUS takes depends on the type of index. Table 46 on page 244 describes these actions.

Restriction

LOADPLUS changes the value of this option to UNIQUECHECK YES if you specify or default to ORDER NO.
**NO**

Specifying this option indicates that you do not want LOADPLUS to check key values during the PRELOAD phase. Using this option eliminates the need to sort and check the indexes in the PRELOAD phase, which can reduce the elapsed time of your load job. When you specify UNIQUECHECK NO and LOADPLUS detects a unique key violation, LOADPLUS deletes the duplicates from the loaded table space and participating indexes but does not place them in the discard data set (SYSDISC).

**REDEFINE**

This option controls whether LOADPLUS deletes and redefines the VSAM data sets for the table space and index spaces that are participating in the load.

LOADPLUS can redefine both user-defined (VCAT-defined) data sets and data sets that are defined in DB2 storage groups (STOGROUP-defined).

**Additional considerations**

Note the following additional information about the REDEFINE option:

- For an SQLAPPLY load, LOADPLUS ignores this option.
- If you are loading a large number of partitions, BMC recommends that you specify REDEFINE NO.
- If you have altered the COMPRESS definition of a participating index, specify REDEFINE YES to enable LOADPLUS to redefine the index.

**Specifying the default**

You can specify the default for the REDEFINE command option in your installation options module by using the REDEFINE installation option. LOADPLUS was shipped with a default value of YES for this option. The command option overrides the default that is in the installation options module.

**YES**

The REDEFINE YES option tells LOADPLUS to delete and redefine the VSAM data set for each DB2 object before loading it. LOADPLUS also defines any additional
VSAM data sets that might be required. The following table describes which DB2 objects that LOADPLUS redefines:

Table 47: DB2 objects redefined with REDEFINE YES

<table>
<thead>
<tr>
<th>Command issued</th>
<th>Objects redefined</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD REPLACE</td>
<td>■ Table spaces (including LOB and XML table spaces, if applicable)</td>
</tr>
<tr>
<td></td>
<td>■ All partitions of participating indexes</td>
</tr>
<tr>
<td>LOAD RESUME YES SHRLEVEL NONE</td>
<td>If you specify or default to INDEX BUILD, all participating indexes</td>
</tr>
<tr>
<td>LOAD RESUME YES PART n</td>
<td>If you specify or default to INDEX BUILD, the corresponding partition of any partitioned index and participating nonpartitioned indexes</td>
</tr>
<tr>
<td>LOAD RESUME YES PART n REPLACE</td>
<td>■ Table partition that you are replacing</td>
</tr>
<tr>
<td></td>
<td>■ Corresponding partition of any partitioned index</td>
</tr>
<tr>
<td></td>
<td>■ If you specify or default to INDEX BUILD, participating nonpartitioned indexes</td>
</tr>
<tr>
<td>LOAD RESUME YES INTO TABLE tableName</td>
<td>All participating indexes</td>
</tr>
<tr>
<td>REPLACE</td>
<td>If you replace all tables in a table space, LOADPLUS also redefines the table space.</td>
</tr>
<tr>
<td>SHRLEVEL CHANGE (except SQLAPPLY load)</td>
<td>Staging data sets</td>
</tr>
<tr>
<td>or SHRLEVEL REFERENCE</td>
<td>If you are loading VCAT-defined objects, you must provide the IDCAMS statements in the SYSIDCIN data set to delete and define the staging data sets. See “Staging data sets” on page 98 for information about naming the staging data sets.</td>
</tr>
</tbody>
</table>

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.

VCAT considerations

For VCAT-defined objects, consider the following information:

■ When you specify PRELOAD PAUSE, you still have the option of deleting and redefining VCAT-defined data sets.
When you specify PRELOAD CONTINUE or PRELOAD LOAD, the only way to delete and redefine VCAT-defined data sets as part of the load is to use this option and provide the SYSIDCIN data set that contains the necessary IDCAMS control statements. Otherwise, LOADPLUS issues message BMC50391E and resets the high-used RBA.

For detailed information about using the SYSIDCIN data set in LOADPLUS, see “SYSIDCIN data set in LOADPLUS” on page 462.

**STOGROUP considerations**

For STOGROUP-defined objects, consider the following information:

- When you specify this option for STOGROUP-defined data sets, do not provide a SYSIDCIN data set.

- For a STOGROUP-defined table space or index space with multiple volumes defined in the storage group, LOADPLUS attempts to reallocate the data set on the volume on which it currently resides if that volume is still defined in the storage group. The order in which LOADPLUS retrieves subsequent volumes from the storage group for the purpose of allocating VSAM data sets is not predictable.

- The redefined table space data sets will have a control interval (CI) size that corresponds with the page size of the assigned buffer pool if the value of your DSVCI system parameter is set to YES.

**NO**

The REDEFINE NO option tells LOADPLUS not to delete and redefine the existing VSAM data sets for the table space and index spaces that are participating in the load. Instead, LOADPLUS issues message BMC50391I, reuses the existing data sets, and resets the high-used RBA.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility as REUSE.

**VCAT considerations**

For VCAT-defined objects, LOADPLUS:

- Extends to another data set, if needed, as long as that data set is already defined
- Does not define any additional data sets

If you specify REDEFINE NO with any load job that uses staging data sets, you must preallocate the staging data sets for VCAT-defined objects. For more information, see “Staging data sets” on page 98.
STOGROUP considerations

For STOGROUP-defined objects, LOADPLUS provides the following functionality:

- For load jobs that use staging data sets, LOADPLUS creates any staging data sets that you do not preallocate and reuses any staging data sets that you do preallocate.

- For multi-data-set objects, LOADPLUS extends to another data set if needed, and creates the data set if it does not exist. When the load completes, LOADPLUS deletes any data set that it did not use.

- The control interval (CI) size for any additional data sets that LOADPLUS creates is based on the value of your DSVCI DB2 system parameter and the page size that is defined in the table space that you are loading.

REUSE

This option is used for compatibility with the IBM DB2 LOAD utility command syntax.

```
REDEFINE  YES
     NO
REUSE
```

If you specify REUSE and REDEFINE YES, LOADPLUS uses the last keyword that it finds in the command string. If you specify REUSE, LOADPLUS functions differently depending on the type of load job that you are running:

- When invoking DSNUTILB:
  - For an SQLAPPLY load, LOADPLUS ignores this option.
  - Otherwise, LOADPLUS passes this option to the DB2 LOAD utility for processing.

- For all other load jobs, LOADPLUS functions as if you specified REDEFINE NO.

IDCDDN

The IDCDDN option allows you to override the default ddname (SYSIDCIN) for the input data set that contains the IDCAMS command statements that LOADPLUS uses to redefine VSAM data sets.
For detailed information about using the SYSIDCIN data set in LOADPLUS, see “SYSIDCIN data set in LOADPLUS” on page 462.

If you use this command option to override the default name in the installation options module, you must also change the ddname in your JCL.

**Specifying the default**

You can specify the default for the IDCDDN command option in your installation options module by using the IDCDDN installation option. LOADPLUS was shipped with a default value of SYSIDCIN for this option. The command option overrides the default that is in the installation options module.

**DELETEFILES**

The DELETEFILES option tells LOADPLUS whether to delete the SORTOUT, SORTWK, SYSUT1, and SYSERR files after the load completes successfully.

**Specifying the default**

You can specify the default for the DELETEFILES command option in your installation options module by using the DELFILES installation option. LOADPLUS was shipped with a default value of YES for this option. The command option overrides the default that is in the installation options module. However, if you are running in a worklist environment, LOADPLUS uses the DELFILES installation option differently.
NO

DELETEFILES NO tells LOADPLUS not to delete any SORTOUT, SORTWK, SYSUT1, or SYSERR files.

**Note**

If your work files are defined with DISP=(any,DELETE,DELETE), LOADPLUS deletes these work files after the load completes even if you specify DELETEFILES NO.

YES

DELETEFILES YES tells LOADPLUS to delete all corresponding SORTOUT, SORTWK, SYSUT1, and SYSERR files after the load completes successfully, regardless of any of the following factors:

- The JCL disposition of these files
- Whether the files were used

LOADPLUS deletes all physical sequential data sets whose ddnames match the SORTOUT, SORTWK, SYSUT1, and SYSERR names or ddname prefixes. This occurs after either the LOAD phase or the COMBINED phase completes successfully.

If the work files are on tape, you can use the TAPEDISP installation option to specify the final disposition of the work files.

**Additional considerations**

The following considerations apply to DELETEFILES YES:

- For a DSNUTILB load, SYSERR data sets are not deleted.

- If you also specify a value for the RETPD or EXPDT option for your dynamically allocated SYSERR data set, LOADPLUS deletes your SYSERR data set only if the expiration date is earlier than the current date, or if the value for RETPD is 0 or blank.

- If the SYSERR data set is allocated as a GDG data set, LOADPLUS deletes only the generation that was created during the current job.

**SYSDISC**

This option tells LOADPLUS whether to delete the SYSDISC file after the load completes successfully.

**Specifying the default**
You can specify the default for the SYSDISC command option in your installation options module by using the second parameter of the DELFILES installation option. LOADPLUS was shipped with a default value of NO for this option. The command option overrides the default that is in the installation options module. However, if you are running in a worklist environment, LOADPLUS uses the DELFILES installation option differently.

**NO**

SYSDISC NO tells LOADPLUS not to delete the SYSDISC file automatically, even if it does not contain any discarded records.

*Note*

If SYSDISC is defined with DISP=(any,DELETE,DELETE), LOADPLUS deletes these data sets after the load completes even if you specify SYSDISC NO.

**YES**

SYSDISC YES tells LOADPLUS to delete the SYSDISC file if it does not contain any discarded records.

*Additional considerations*

The following considerations apply to SYSDISC YES:

- For a DSNUTILB load, LOADPLUS ignores this option.

- If you also specify a value for the RETPD or EXPDT option for your dynamically allocated SYSDISC data set, LOADPLUS deletes your SYSDISC data set only if the expiration date is earlier than the current date, or if the value for RETPD is 0 or blank.

- If the SYSDISC data set is allocated as a GDG data set, LOADPLUS deletes only the generation that was created during the current job.

**ORIGINALDISP**

ORIGINALDISP allows you to specify the action that you want LOADPLUS to take after it has successfully renamed or switched the staging data sets and completed the load job.
This option is valid only if SHRLEVEL CHANGE (except an SQLAPPLY load) or SHRLEVEL REFERENCE is in effect.

For information about staging data sets, see “Staging data sets” on page 98.

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.

Specifying the default

You can specify the default for the ORIGINALDISP command option in your installation options module by using the ORIGDISP installation option. LOADPLUS was shipped with a default value of DELETE for this option. The command option overrides the default that is in the installation options module.

DELETE

This option tells LOADPLUS to delete the original data sets.

RENAME

This option tells LOADPLUS to rename the original data sets to the staging data set names. LOADPLUS changes the OLD in the OLDDBC and OLDDBD node of the cluster and data component names to BMC.

This option allows you to preserve the space that was initially allocated for the original data sets by renaming them to the staging data set names. The renamed data sets are then ready to use as the staging data sets in a subsequent load process.

LOADPLUS ignores the ORIGINALDISP RENAME option if either of the following conditions exists:

- You specify YES for the FASTSWITCH option.
- You specify ZPARM for the FASTSWITCH installation option and the DB2 ZPARM value is YES.

In these cases, no rename is needed. The data sets keep their original names, and the space is preserved for use in a subsequent load job. For more details about the rename process, see “Staging data sets and the data set rename process” on page 100.

FASTSWITCH

The FASTSWITCH option determines the action that LOADPLUS takes in the UTILTERM phase regarding the staging data sets.
This option is valid only if SHRLEVEL CHANGE (except an SQLAPPLY load) or SHRLEVEL REFERENCE is in effect. For more information about FASTSWITCH processing, see “Staging data sets and the FASTSWITCH process” on page 102.

When invoking DSNUTILB, the value in DSNZPARM is used for this option.

**Specifying the default**

You can specify the default for the FASTSWITCH command option in your installation options module by using the FASTSWITCH installation option. LOADPLUS was shipped with a default value of NO for this option. The command overrides the default that is in the installation options module.

**YES**

YES tells LOADPLUS to bypass the VSAM rename process and points the DB2 catalog to the staging data sets.

**Restriction**

When loading clone tables or base objects that participate (or have participated) in a clone relationship, LOADPLUS changes FASTSWITCH YES to FASTSWITCH NO.

**Additional considerations**

Consider the following information before using the FASTSWITCH process:

- BMC recommends using the FASTSWITCH process when you are loading objects with more than 200 data sets, such as a segmented table space that contains many tables that each have an index.

- The FASTSWITCH process requires updates to the DB2 catalog and directory. In some environments, LOADPLUS might have difficulty acquiring the necessary locks, which results in contention or deadlocks. Minimizing use of the FASTSWITCH process can reduce this contention and allow the process to be more effective when it is needed. Frequently reorganizing the DB2 catalog and directory can also help prevent contention.

**NO**

NO tells LOADPLUS to rename the staging data sets to the names of the original data sets.
SWITCHTIME

SWITCHTIME allows you to tell LOADPLUS when you want the rename or FASTSWITCH process to begin for your staging data sets.

This option is valid only if SHRLEVEL CHANGE (except an SQLAPPLY load) or SHRLEVEL REFERENCE is in effect.

Restrictions

The following restrictions apply to SWITCHTIME:

- When invoking DSNUTILB, LOADPLUS ignores this option.
- If you specify SWITCHTIME and EXCHANGE, LOADPLUS uses the last keyword that it finds in the command string.

NOW

SWITCHTIME NOW (the default) tells LOADPLUS to begin the rename or FASTSWITCH process as soon as the load process is complete.

PAUSE

SWITCHTIME PAUSE tells LOADPLUS to pause after the load process is complete.

Use this option to control the outage incurred during the rename or FASTSWITCH process. By specifying PAUSE, you can schedule the UTILTERM phase to minimize the impact on data availability.

Restarting

You can restart the utility at the beginning of the UTILTERM phase by specifying RESTART or RESTART(PHASE) on your EXEC statement. Note the following information about restarting your job:

- You do not need to change the SWITCHTIME option on the LOAD command. When restarting a SWITCHTIME PAUSE job, LOADPLUS ignores the PAUSE specification.
When you restart a SWITCHTIME PAUSE job, the object must be in the same status that it was in during the original load job.

WTOR

SWITCHTIME WTOR tells LOADPLUS to issue the following message to the MVS system console after the load process is complete, and to wait for a reply:

```
BMC5156A  JOB jobName, STEP stepName, FOR TABLESPACE DBName.tableSpaceName
REACHED SWITCHTIME. REPLY (G)O OR (P)AUSE
```

Specify G (or GO) to tell LOADPLUS to start the rename or FASTSWITCH process now. Specify P (or PAUSE) to tell LOADPLUS to pause. When you specify P in response to this message, LOADPLUS functions as if you specified SWITCHTIME PAUSE.

EXCHANGE

EXCHANGE tells LOADPLUS that you want to exchange the base and clone objects after the load process completes.

If you specify EXCHANGE without a value, LOADPLUS assumes EXCHANGE NOW.

Restrictions

The following restrictions apply to the EXCHANGE option:

- LOADPLUS terminates if you specify this option when loading an object other than a clone table.
- When invoking DSNUTILB, LOADPLUS ignores this option.
- If you specify SWITCHTIME and EXCHANGE, LOADPLUS uses the last keyword that it finds in the command string.

Additional considerations

The following considerations apply to the EXCHANGE option:
For load jobs that use staging data sets, LOADPLUS completes the renaming process before starting the exchange process.

If you are running jobs to load both the base and clone objects, and you specify EXCHANGE, run the jobs sequentially to avoid unexpected results.

**NOW**

EXCHANGE NOW tells LOADPLUS to begin the exchange process as soon as the load process is complete.

**PAUSE**

EXCHANGE PAUSE tells LOADPLUS to pause after the load process is complete. Use this option to control the timing of the exchange.

**Restarting**

You can restart the utility at the beginning of the UTILTERM phase by specifying RESTART or RESTART(PHASE) on your EXEC statement. Note the following information about restarting your job:

- You do not need to change the EXCHANGE option on the LOAD command. When restarting an EXCHANGE PAUSE job, LOADPLUS ignores the PAUSE specification.
- When you restart an EXCHANGE PAUSE job, the object must be in the same status that it was in during the original load job.

**WTOR**

EXCHANGE WTOR tells LOADPLUS to issue the following message to the MVS system console after the load process is complete, and to wait for a reply:

```
BMC51563A  JOB jobName, STEP stepName, FOR TABLESPACE DBName.tableSpaceName
REACHED EXCHANGE. REPLY (G)O OR (P)AUSE
```

Specify G (or GO) to tell LOADPLUS to start the exchange process now. Specify P (or PAUSE) to tell LOADPLUS to pause. When you specify P in response to this message, LOADPLUS functions as if you specified EXCHANGE PAUSE.

**NLPCTFREE**

NLPCTFREE specifies the percentage of each nonleaf index page to reserve as free space when LOADPLUS builds the indexes.
This option is valid only if LOADPLUS is building an index.

**Restriction**

When invoking DSNUTILB, LOADPLUS ignores this option.

**STANDARD**

STANDARD (the default) tells LOADPLUS to use the value that was provided in the DB2 PCTFREE option when the index was created. Like DB2, LOADPLUS leaves up to 10 percent of a nonleaf page free. If you specify a value greater than 10, only 10 percent is left free.

**BMC**

This option tells LOADPLUS to honor the DB2 PCTFREE value in the DB2 catalog, even if the value is greater than 10.

**integer**

This value identifies the percentage of each nonleaf index page to reserve as free space. You can specify an integer value of 0 through 99.

**ANALYZE**

ANALYZE gathers information about the table space and provides estimated data set sizes.

LOADPLUS estimates size for the following data sets:
- Load (SORTOUT)
- Work (SYSUT1)
- Sort work (SORTWK)
- Discard (SYSDISC)
- Error (SYSERR)
- Image copy (BMCCPY, BMCCPZ, BMCRCY, and BMCRCZ)

You must either enable automated file size estimation (AUTOENUMROWS=YES or ENUMROWS AUTO), or specify a numeric value for the ENUMROWS option to allow ANALYZE processing to estimate the required space. If you specify ANALYZE PAUSE or ANALYZE ONLY without an ENUMROWS specification, LOADPLUS terminates. For more information, see “ENUMROWS” on page 229.

LOADPLUS provides estimates in both kilobytes and cylinders by device type. LOADPLUS calculates the data set size based on two values: cardinality (the number of rows) and average row length. To obtain the cardinality, LOADPLUS either samples or scans (depending on what you specify) one index per participating table. LOADPLUS determines the average row length from information in the DB2 catalog. The following table provides additional information.

LOADPLUS scans or samples all data sets of a multi-data-set index. LOADPLUS multitasks its analysis, scanning or sampling one data set per task.

Table 48: How ANALYZE options determine cardinality and average row length estimation

<table>
<thead>
<tr>
<th>ANALYZE keywords</th>
<th>How LOADPLUS determines cardinality</th>
<th>How LOADPLUS determines average row length</th>
</tr>
</thead>
<tbody>
<tr>
<td>(ANALYZE not specified)</td>
<td>If you enable dynamic allocation, LOADPLUS defaults to ANALYZE and issues message BMC51424I. If you specify LOAD RESUME YES and ENUMROWS AUTO is in effect, LOADPLUS defaults to ANALYZE. Otherwise, LOADPLUS does not perform an analysis.</td>
<td></td>
</tr>
<tr>
<td>ANALYZE</td>
<td>If running LOAD RESUME YES INDEX BUILD (with the exceptions noted in this topic), LOADPLUS decides whether to sample or scan the index leaf pages. Otherwise, LOADPLUS does not determine cardinality, but uses the number of new rows to determine work data set size.</td>
<td>LOADPLUS obtains this information from the DB2 catalog, basing it on half the length of any VARCHAR columns in the table. LOADPLUS does not consider compression in the calculations.</td>
</tr>
<tr>
<td>ANALYZE PAUSE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANALYZE ONLY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANALYZE keywords</td>
<td>How LOADPLUS determines cardinality</td>
<td>How LOADPLUS determines average row length</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ANALYZE SAMPLE</td>
<td>If running LOAD RESUME YES INDEX BUILD (with the exceptions noted in this topic), LOADPLUS samples the index space. Otherwise, LOADPLUS does not determine cardinality, but uses the number of new rows to determine work data set size.</td>
<td>LOADPLUS obtains this information from the DB2 catalog, basing it on half the length of any VARCHAR columns in the table. LOADPLUS does not consider compression in the calculations.</td>
</tr>
<tr>
<td>ANALYZE PAUSE SAMPLE</td>
<td>If running LOAD RESUME YES INDEX BUILD (with the exceptions noted in this topic), LOADPLUS scans the most appropriate index. Otherwise, LOADPLUS does not determine cardinality, but uses the number of new rows to determine work data set size.</td>
<td>LOADPLUS obtains this information from the DB2 catalog, basing it on half the length of any VARCHAR columns in the table. LOADPLUS does not consider compression in the calculations.</td>
</tr>
<tr>
<td>ANALYZE ONLY SAMPLE</td>
<td>If running LOAD RESUME YES INDEX BUILD (with the exceptions noted in this topic), LOADPLUS scans the most appropriate index. Otherwise, LOADPLUS does not determine cardinality, but uses the number of new rows to determine work data set size.</td>
<td>LOADPLUS obtains this information from the DB2 catalog, basing it on half the length of any VARCHAR columns in the table. LOADPLUS does not consider compression in the calculations.</td>
</tr>
</tbody>
</table>

**Restriction**

This option is not valid when LOADPLUS invokes DSNUTILB. LOADPLUS issues message BMC50178E and terminates.

**Additional considerations**

Note the following considerations when specifying an ANALYZE option:

- LOADPLUS determines cardinality only when running LOAD RESUME YES INDEX BUILD (because the indexes will be built by using existing keys). LOADPLUS does not determine cardinality in the following cases:
  - If you specify LOAD RESUME NO or LOAD REPLACE
  - If you specify LOAD RESUME YES INTO TABLE tableName REPLACE
  - If you specify LOAD RESUME YES INDEX UPDATE
  - If you specify LOAD RESUME YES PART n REPLACE and no nonpartitioned indexes are participating in the load
  - If you are running an SQLAPPLY load
  - For partitions or tables that are not part of the load, unless they have keys in a nonpartitioned index that the load will build

- During the ANALYZE phase, LOADPLUS does not stop the index space and associated table space that are participating in the load. See “ANALYZE phase in
LOADPLUS” on page 690 for performance considerations when using this option.

- When either of the following conditions exists, LOADPLUS determines whether to use sampling or scanning. LOADPLUS performs a full analysis and continues processing, using the information from the ANALYZE phase to dynamically allocate work files.
  - You specify ANALYZE without a value.
  - You do not specify ANALYZE, but you do enable dynamic data set allocation.

**PAUSE**

If you specify ANALYZE PAUSE, LOADPLUS generates a report and ends the processing after the ANALYZE phase completes.

You can use the output of the ANALYZE phase to specify the number and allocations of the work data sets. LOADPLUS leaves the utility ID active so that you can restart the load at the next phase by specifying the RESTART or RESTART(PHASE) parameter on your EXEC statement. You do not need to change the ANALYZE option on the LOAD command. When restarting an ANALYZE PAUSE job, LOADPLUS ignores the PAUSE specification. For an example of a job that uses this option, see “Example 8 LOAD REPLACE with ANALYZE PAUSE” on page 538.

**Note**

LOADPLUS does not report estimates for the LOB and XML copy data sets. These data sets must always be dynamically allocated.

**ONLY**

If you specify ANALYZE ONLY, LOADPLUS generates a report and terminates after the ANALYZE phase. You cannot restart the load job. However, you can use the output of the ANALYZE phase to specify the number and allocations of the work data sets for subsequent load jobs.

**Note**

LOADPLUS does not report estimates for the LOB and XML copy data sets. These data sets must always be dynamically allocated.

**SAMPLE**

SAMPLE tells LOADPLUS to determine the estimated cardinality by reading a subset of pages from the most appropriate index. LOADPLUS determines the most appropriate index based on key length and the number of data sets in the index.
**Note**

If LOADPLUS attempts to sample a data set that contains fewer than 100 pages, LOADPLUS actually scans the data set instead of sampling it.

---

**SCAN**

SCAN tells LOADPLUS to determine the exact cardinality by reading every leaf page in the most appropriate index. LOADPLUS determines the most appropriate index based on key length and the number of data sets in the index.

---

**SYNC**

LOADPLUS writes records to the BMCSYNC table that indicate the number of 1-KB rows that LOADPLUS processed during the LOAD phase or the COMBINED phase.

\[ \text{SYNC integer} \]

You can use this information to determine how far the load job has progressed.

By default, LOADPLUS writes BMCSYNC records only after loading the last row or key in a table space, index, or partition. If you want records written more often, specify an integer with the SYNC option to identify the number of 1-KB rows that you want LOADPLUS to process between writing to the BMCSYNC table.

**Additional considerations**

Note the following additional information about the SYNC option:

- LOADPLUS records monitoring and restart sync points in the BMCSYNC table as the job progresses. You can issue an SQL statement to query this table to determine how far the load has progressed and the status of the objects that you are loading.

- The SYNC option does not control the sync points that are used to restart LOADPLUS. The restart sync points are established only after LOADPLUS loads the last row or key in a table space, index, or partition.

- LOADPLUS ignores this option when either of the following conditions exists:
  — You are running an SQLAPPLY load.
  — LOADPLUS is invoking DSNUTILB.
ENFORCE

ENFORCE tells LOADPLUS whether to check for violations of DB2-defined constraints.

For more information about how LOADPLUS handles constraint checking, see “Referential and check constraints” on page 146.

Additional considerations

The following considerations apply to the ENFORCE option:

- For an SQLAPPLY load:

  — If LOADPLUS invokes DSNUTILB, LOADPLUS passes ENFORCE CONSTRAINTS regardless of the value that you specify.

  — For all other SQLAPPLY load jobs, LOADPLUS ignores this option because DB2 checks all constraints during apply processing.

- If the ENFORCE option that is in effect causes LOADPLUS to not check for all referential and check constraint violations defined on the table space, LOADPLUS places the affected table spaces in CHKP status if CHECKPEND YES is in effect. For more information about when LOADPLUS sets and resets CHKP status, see “Considerations for CHKP status” on page 150.

Specifying the default

You can specify the default for the ENFORCE command option in your installation options module by using the ENFORCE installation option. LOADPLUS was shipped with a default value of CHECK for this option. The command option overrides the default that is in the installation options module.
**CONSTRAINTS**

When you specify CONSTRAINTS, LOADPLUS checks the rows to be loaded for violations of both DB2-defined check constraints and referential constraints, if any constraints exist. LOADPLUS discards any rows that violate constraints.

To check referential constraints, you must have XBM or SUF installed and available to LOADPLUS. For information about additional requirements and restrictions when checking referential constraints, see “Referential constraints and informational referential constraints” on page 146.

**Additional considerations**

Note the following additional information about ENFORCE CONSTRAINTS:

- If LOADPLUS cannot enforce all constraints defined on the table that you are loading and CHECKPEND YES is in effect, LOADPLUS sets CHKP status on that table.

- If you specify FORMAT BMCUNLOAD, LOADPLUS does not enforce check constraints and issues message BMC50138I. LOADPLUS sets CHKP status if check constraints are defined on the table that you are loading and CHECKPEND YES is in effect.

- When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing unless you also specify INFORI YES.

**INFORI**

This option tells LOADPLUS whether to enforce informational referential constraints when enforcing all constraints.

**NO**

INFORI NO tells LOADPLUS to ignore informational referential constraints when enforcing all constraints.

**YES**

INFORI YES tells LOADPLUS to enforce informational referential constraints in the same way as other referential constraints.

**Restriction**

This option is not valid when LOADPLUS invokes DSNUTILB:

- For an SQLAPPLY load, LOADPLUS passes ENFORCE CONSTRAINTS.
Otherwise, LOADPLUS issues message BMC50178E and terminates.

Additional considerations

Be aware of the following information when you specify INFORI YES:

■ LOADPLUS discards any violating rows, just as it does for other referential constraints.

■ If LOADPLUS sets CHKP status on a table because it enforced informational referential constraints on the parent table, you might not be able to use the standard methods for resetting CHKP status. Therefore, if you are enforcing only informational constraints, BMC recommends that you also specify CHECKPEND NO.

For more information about resetting CHKP status, see “Restoring integrity” on page 151.

RI

When you specify RI, LOADPLUS checks the rows to be loaded for violations of DB2-defined referential constraints only, if any constraints exist. LOADPLUS discards any violating rows.

To check referential constraints, you must have XBM or SUF installed and available to LOADPLUS. For information about additional requirements and restrictions when checking referential constraints, see .

Restriction

This option is not valid when LOADPLUS invokes DSNUTILB:

■ For an SQLAPPLY load, LOADPLUS passes ENFORCE CONSTRAINTS.
■ Otherwise, LOADPLUS issues message BMC50178E and terminates.

INFORI

This option tells LOADPLUS whether to enforce informational referential constraints when enforcing other referential constraints.

NO

INFORI NO tells LOADPLUS to ignore informational referential constraints when enforcing referential constraints.

YES

INFORI YES tells LOADPLUS to enforce informational referential constraints in the same way as other referential constraints.
**Additional considerations**

Be aware of the following information when you specify INFORI YES:

- LOADPLUS discards any violating rows, just as it does for other referential constraints.

- If LOADPLUS sets CHKP status on a table because it enforced informational referential constraints on the parent table, you might not be able to use the standard methods for resetting CHKP status. Therefore, if you are enforcing only informational constraints, BMC recommends that you also specify CHECKPEND NO.

  For more information about resetting CHKP status, see “Restoring integrity” on page 151.

**CHECK CONSTRAINTS**

When you specify CHECK CONSTRAINTS, LOADPLUS checks the rows to be loaded for violations of DB2-defined check constraints only, if any constraints exist. LOADPLUS discards any violating rows.

**Restrictions**

The following restrictions apply to the ENFORCE CHECK CONSTRAINTS option:

- If you specify FORMAT BMCUNLOAD, LOADPLUS ignores this option and issues message BMC50138I. LOADPLUS sets CHKP status if there are check constraints on the table that you are loading and CHECKPEND YES is in effect.

- This option is not valid when LOADPLUS invokes DSNUTILB:
  - For an SQLAPPLY load, LOADPLUS passes ENFORCE CONSTRAINTS.
  - Otherwise, LOADPLUS issues message BMC50178E and terminates.

**NO**

Specify NO when you do not want to verify check constraints or referential constraints during LOADPLUS processing. When you specify NO, LOADPLUS loads all rows regardless of whether they violate check constraints or referential constraints. If CHECKPEND YES is in effect, LOADPLUS automatically sets the object status to CHKP.

When invoking DSNUTILB,

- For an SQLAPPLY load, LOADPLUS passes ENFORCE CONSTRAINTS.

- Otherwise, LOADPLUS passes this option to the DB2 LOAD utility for processing.
CHECKPEND

CHECKPEND tells LOADPLUS whether to set dependent table spaces to CHECK pending (CHKP) status.

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.

Specifying the default

You can specify the default for the CHECKPEND command option in your installation options module by using the CHEKPEND installation option. LOADPLUS was shipped with a default value of YES for this option. With the following exception, the command option overrides the default that is in the installation options module.

If the value of the CHEKPEND installation option includes the ENFORCE keyword, you cannot override the installation option with the CHECKPEND command option.

YES

After loading, LOADPLUS sets CHKP status if appropriate and completes with return code 4. When CHKP status is set, ensure that referential integrity and table check integrity have not been violated by performing one of the tasks described in “Restoring integrity” on page 151.

If the value for the CHEKPEND installation option is (NO,ENFORCE), specifying CHECKPEND YES causes LOADPLUS to issue message BMC50115E and terminate.

For an SQLAPPLY load, LOADPLUS ignores this option.

NO

After loading, LOADPLUS does not set CHKP status and the job completes with return code 0.

If the value for the CHEKPEND installation option is (YES,ENFORCE), specifying CHECKPEND NO causes LOADPLUS to issue message BMC50115E and terminate.
WARNING
If you are considering specifying CHECKPEND NO, ensure that you understand the risks and ramifications to the integrity of your RI data structure.

ZIIP

The ZIIP option tells LOADPLUS whether to attempt to use IBM z Integrated Information Processors (zIIPs).

LOADPLUS 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 LOADPLUS jobs.

When invoking DSNUTILB, LOADPLUS ignores this option.

Specifying the default

You can specify the default for the ZIIP command option in your installation options module by using the ZIIP installation option. LOADPLUS was shipped with a default value of ENABLED for this option. The ZIIP command option overrides the default that is in the installation options module.

ENABLED

ZIIP ENABLED tells LOADPLUS 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 LOADPLUS, you must:

- Have an installed authorized version of XBM or SUF
- Start and maintain an XBM subsystem in your environment

Note
You can specify a particular XBM subsystem to use by specifying a value for the XBMID installation or command option. For more information, see “XBMID” on page 269.

- Have a zIIP available in your environment
For more information, see the following references:

- For XBM or SUF requirements and considerations, see “XBM and SUF considerations in LOADPLUS” on page 160.
- For the steps to follow to enable zIIP processing, see “Using XBM or SUF to enable zIIP processing” on page 161.
- For more information about the XBM component that enables the use of zIIPs, see the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide.

For information about how to determine zIIP usage, view the Quick course "EXTENDED BUFFER MANAGER for DB2 - Measuring zIIP Usage."

**DISABLED**

ZIIP DISABLED tells LOADPLUS to not attempt to use zIIP processing.

**XBMID**

Specify XBMID to identify a specific active XBM subsystem to use for snapshot processing and zIIP processing in LOADPLUS.

```
XBMID ssid
```

The variable ssid (subsystem ID) is the unique identifier that you specified when you installed XBM or SUF. If you are using XBM or SUF in a DB2 data sharing environment, you can use the value of the XBMGROUP parameter in place of the ssid. The XBMGROUP is the name of the cross-system coupling facility (XCF) group that is defined to the XBM subsystem, and its default value is XBMGROUP.

If you specify an XBM subsystem, it must be:

- Available
- At a supported maintenance level
- Enabled for the required function

If you do not specify an XBM subsystem (either here or with the XBMID installation option), LOADPLUS automatically searches for an XBM subsystem that meets the same criteria. If you have multiple subsystems that meet these criteria, LOADPLUS can use any one of these subsystems.
For more information about using XBM or SUF with LOADPLUS, see “XBM and SUF considerations in LOADPLUS” on page 160.

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.

Specifying the default

You can specify the default for the XBMID command option in your installation options module by using the XBMID installation option. The command option overrides the default that is in the installation options module.

LOG

LOADPLUS provides this option for compatibility with the IBM DB2 LOAD utility command syntax.

When invoking DSNUTILB, LOADPLUS passes this option to the DB2 LOAD utility.

YES

Except when invoking DSNUTILB, LOADPLUS functions as follows:

- For an SQLAPPLY load, LOADPLUS treats this option as a comment.
- For all other load types, LOADPLUS issues message BMC50110E and terminates.

NO

Except when invoking DSNUTILB, LOADPLUS functions as follows:

- For an SQLAPPLY load, LOADPLUS issues message BMC50115E and terminates.
- For all other load types, if you specify this option without the NOCOPYPEND keyword, LOADPLUS treats it as a comment.
NOCOPYPEND

LOADPLUS provides this option for compatibility with the IBM DB2 LOAD utility command syntax. When you specify this option, LOADPLUS functions as if you specified COPY NO COPYPEND NO.

Note
If the value for the COPYPEND installation option is (YES,ENFORCE), specifying NOCOPYPEND causes LOADPLUS to issue message BMC50115E and terminate.

PREFORMAT

The PREFORMAT option tells LOADPLUS whether to preformat the unused pages of the data set.

If you specify PREFORMAT without a value, LOADPLUS assumes PREFORMAT YES.

LOADPLUS also provides PREFORMAT support at the partition level. See “PART” on page 310.

Specifying the default

You can specify the default for the PREFORMAT option in your installation options module by using the PREFORMAT installation option. LOADPLUS was shipped with a default value of NO for this option. The command option overrides the default that is in the installation options module.

YES

If you specify PREFORMAT YES, LOADPLUS preformats the unused portion of the data set, writing full pages that have been initialized with zeros up to the high-allocated RBA of the table space and index spaces. Preformatting occurs after LOADPLUS loads the data and builds the indexes.

For an SQLAPPLY load, LOADPLUS ignores this option and issues message BMC50109I.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility as PREFORMAT.
NO

If you specify PREFORMAT NO, LOADPLUS does not preformat the unused pages.

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.

CENTURY

The CENTURY option specifies the 100-year range that determines the century for DATE and TIMESTAMP external formats that contain two-digit year values.

The first four-digit year value must be less than the second four-digit year. You must specify both values and these values must span 100 years.

You can also specify CENTURY with the field specification option to apply it to a particular DATE or TIMESTAMP column. For more information, see “DATE” on page 349.

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.

Specifying the default

You specify the default for the CENTURY command option in your installation options module by using the CENTURY installation option. LOADPLUS was shipped with a default value of (1950,2049) for this option. The command option overrides the default that is in the installation options module.

\[(ccyy,ccyy)\]

Any two-digit year in the input data that lies between the first yy specification and 99 is prefixed with the first cc value to create a four-digit year. Any two-digit year in the input data that lies between 00 and the second yy specification is prefixed with the second cc value to create a four-digit year.

For example, if you specify CENTURY(1950,2049), LOADPLUS places 19 in front of each two-digit year with a value 50 through 99, and places 20 in front of each two-digit year with a value 00 through 49. The date 99/12/31 becomes 1999/12/31 and 00/12/31 becomes 2000/12/31.
**IMPLICIT_TZ**

The IMPLICIT_TZ option allows you to specify the implicit time zone to use when loading a TIMESTAMP WITH TIME ZONE column with timestamp values that do not contain a time zone.

```
IMPLICIT_TZ 'timeZoneString'
```

Specify a time zone, within quotation marks, as the difference between local time and Coordinated Universal Time (UTC). You can specify a value from -12:59 through +14:00. If you do not specify a value for the IMPLICIT_TZ option, LOADPLUS uses the DSNHDECP IMPLICIT_TIMEZONE value.

When invoking DSNUTILB, LOADPLUS passes this option to the DB2 LOAD utility.

**IDCACHE**

The IDCACHE option allows you to specify the size of the cache that LOADPLUS uses when generating values for an identity column.

```
IDCACHE integer
```

Specify the size as the number of values to reserve. LOADPLUS reserves at least one cache for each read task. The valid values for this option are 1 through 100000.

**Note**

This cache is a temporary cache that LOADPLUS uses. This option does not affect the cache that you specified when defining your identity column.

LOADPLUS reserves this cache of numbers in SYSIBM.SYSSEQUENCES. If you specify a smaller size for this cache, LOADPLUS accesses and updates SYSIBM.SYSSEQUENCES more frequently, which can reduce the performance of your load job. However, if you specify a larger size for this cache, there is a greater possibility of gaps in your identity column values and a higher risk of running out of numbers. In addition, if your identity column is defined with CYCLE YES, a larger cache size introduces a greater likelihood of generating duplicate values.

**Restrictions**
LOADPLUS ignores this option in either of the following cases:

- You are running an SQLAPPLY load.
- LOADPLUS is invoking DSNUTILB.

**Specifying the default**

You can specify the default for the IDCACHE command option in your installation options module by using the IDCACHE installation option. LOADPLUS was shipped with a default value of 1000 for this option. The command option overrides the default that is in the installation options module.

**UPDATEMAXA**

The UPDATEMAXA option tells LOADPLUS whether to update the MAXASSIGNEDVAL column of SYSIBM.SYSSEQUENCES when loading identity column values from an input file.

LOADPLUS validates the input file values against the range that is defined on the identity column and updates the value of MAXASSIGNEDVAL based on this range and the current value of MAXASSIGNEDVAL.

**Note**

When generating all identity column values, LOADPLUS automatically updates MAXASSIGNEDVAL, regardless of the value for the UPDATEMAXA option.

For details about how LOADPLUS updates the MAXASSIGNEDVAL column, see “How LOADPLUS updates MAXASSIGNEDVAL when loading identity columns” on page 111.

**Restriction**

When invoking DSNUTILB, LOADPLUS ignores this option.

**Specifying the default**

You can specify the default for the UPDATEMAXA command option in your installation options module by using the UPDMAXA installation option.
LOADPLUS was shipped with a default value of NO for this option. The command option overrides the default that is in the installation options module.

**YES**

UPDATEMAXA YES tells LOADPLUS to update the MAXASSIGNEDVAL field of SYSIBM.SYSSEQUENCES when loading identity column values from an input file.

To enable LOADPLUS to update MAXASSIGNEDVAL when you specify UPDATEMAXA YES, one of the following authorization IDs must have ALTER privileges on the table that you are loading. The UPDMAXA_AUTHID installation option controls which ID must have these privileges:

- User ID of the job owner, when UPDMAXA_AUTHID=USER
- INSTALL SYSADM, when UPDMAXA_AUTHID=INSTALLSYSADM

**NO**

UPDATEMAXA NO tells LOADPLUS not to update the MAXASSIGNEDVAL field of SYSIBM.SYSSEQUENCES when loading identity column values from an input file.

**IDERROR**

The IDERROR option tells LOADPLUS what action to take when encountering a generated identity column value that is outside the range that is defined on the column.

![Diagram](image)

*Specifying the default*

You can specify the default for the IDERROR command option in your installation options module by using the IDERROR installation option. LOADPLUS was shipped with a default value of DISCARD for this option. The command option overrides the default that is in the installation options module.

**DISCARD**

If you specify IDERROR DISCARD, LOADPLUS discards any generated identity column values that are outside the range that is defined on the column.


**Restriction**

This option is not valid when LOADPLUS invokes DSNUTILB. LOADPLUS issues message BMC50178E and terminates.

**FAIL**

If you specify IDERROR FAIL, LOADPLUS terminates as soon as it encounters a generated identity column value that is outside the range that is defined on the column.

**Restrictions**

The following restrictions apply to IDERROR FAIL:

- For an SQLAPPLY load, LOADPLUS ignores this option and always discards generated identity column values that are outside the range that is defined on the column.
- When invoking DSNUTILB, LOADPLUS ignores this option.

**IDENTITYOVERRIDE**

The IDENTITYOVERRIDE option tells LOADPLUS whether to load identity column values from an input file when the identity column is defined as GENERATED ALWAYS.

If you specify IDENTITYOVERRIDE without a value, LOADPLUS assumes IDENTITYOVERRIDE YES.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility.

**Restriction**

For an SQLAPPLY load, LOADPLUS ignores this option.
YES

IDENTITYOVERRIDE YES tells LOADPLUS to load identity column values from the input file when the identity column is defined as GENERATED ALWAYS.

To update the MAXASSIGNEDVAL column of the SYSIBM.SYSSEQUENCES table, you must also specify UPDATEMAX YES

NO

IDENTITYOVERRIDE NO tells LOADPLUS not to load identity column values from the input file when the identity column is defined as GENERATED ALWAYS.

If the table that you are loading contains an identity column that is defined as GENERATED ALWAYS, LOADPLUS operates as follows:

- If you do not include a field specification for the identity column, LOADPLUS generates identity column values.
- If you include a field specification for the identity column, LOADPLUS issues message BMC50347E and terminates.

DRAIN_WAIT

The DRAIN_WAIT option specifies the drain timeout value to use.

If it cannot drain all of the objects within the time period specified by DRAIN_WAIT, LOADPLUS completes the following process:

1. Releases the drains that it has obtained so far
2. Waits the length of time that you specify in the RETRY_DELAY command option (or DRNDELAY installation option)
3. Tries again to drain the objects for the number of times that you specify in the RETRY command option (or DRNRETRY installation option)
Note
The FORCE option tells LOADPLUS whether to cancel DB2 threads that might prevent a successful drain and, if so, when to cancel them. Depending on the FORCE option that you specify, the drain process might differ from the process described here.

Restriction
For a DSNUTILB load, LOADPLUS ignores this option.

Specifying the default
You can specify the default for the DRAIN_WAIT command option in your installation options module by using the DRNWAIT installation option. LOADPLUS was shipped with a default value of NONE for this option. The command option overrides the default that is in the installation options module.

NONE

If you specify NONE, the drain request that LOADPLUS issues times out immediately if the drain cannot acquire the lock. NONE prevents any application transactions from being queued during the drain process. BMC recommends that you specify NONE in high-transaction environments.

UTIL

UTIL tells LOADPLUS to use the standard DB2 utility timeout value defined in DSNZPARMS for your site (IRLMRWT multiplied by UTIMOUT). The wait time applies to each object involved in the load.

SQL

If you specify SQL, LOADPLUS uses the standard SQL timeout value (IRLMRWT) as the drain timeout value. The wait time applies to each object involved in the load.

integer

Specify any integer value from 0 through 1800, as follows:

- 0 is equivalent to the value UTIL.
- 1 through 1800 specifies the number of seconds to wait to obtain the drain for each drain retry before timing out.
RETRY

The RETRY option specifies the maximum number of times that you want the utility to attempt to obtain a drain before it terminates.

The number of attempts can range from 0 through 255.

Specifying the default

You can specify the default for the RETRY command option in your installation options module by using the DRNRETRY installation option. The utility was shipped with a default value of 255 for this option. The command overrides the default that is in the installation options module.

RETRY_DELAY

After a drain times out, the RETRY_DELAY option specifies the minimum number of seconds that you want the utility to wait before it tries again to obtain the drain.

The number of seconds can range from 1 through 1800.

Specifying the default

You can specify the default for the RETRY_DELAY command option in your installation options module by using the DRNDELAY installation option. The utility was shipped with a default value of 1 for this option. The command overrides the default that is in the installation options module.

FORCE

The FORCE option specifies whether to cancel DB2 threads that might prevent a drain process from completing.
When invoking DSNUTILB, LOADPLUS ignores this option.

**Specifying the default**

You can specify a default for the FORCE command option in your installation options module by using the FORCE installation option. LOADPLUS was shipped with a default value of NONE for this option. The FORCE command option overrides the default that is in the installation options module.

**NONE**

FORCE NONE tells LOADPLUS not to cancel DB2 threads that might prevent the drain process from completing.

**READERS**

FORCE READERS tells LOADPLUS to cancel read claimers at the point specified by the FORCE_AT option.

**FORCE_AT**

The FORCE_AT option tells LOADPLUS at which point during the drain process to cancel DB2 threads that might prevent the drain process from completing.

**Specifying the default**

You can specify a default for the FORCE_AT command option in your installation options module by using the first parameter of the FORCE_AT installation option. LOADPLUS was shipped with a default value of START
for this option. The FORCE_AT command option overrides the default that is in the installation options module.

**Additional consideration**

The cancelation process includes a small amount of time to issue the cancel commands, plus any time that DB2 needs to roll back the canceled transactions. If the drain wait time (controlled by the DRAIN_WAIT option) is shorter than the total cancelation process time, the drain fails. FORCE_AT START and FORCE_AT RETRY allow for additional drain attempts after the start of the cancelation process; therefore, these options increase the likelihood that the drain will succeed when a short drain wait time is in effect. (Alternatively, you can increase the drain wait time.)

**START**

FORCE_AT START tells LOADPLUS to start canceling threads when the drain request begins.

**RETRY**

FORCE_AT RETRY tells LOADPLUS to start canceling threads the first time the drain process times out and LOADPLUS attempts to retry the drain.

**LASTRETRY**

FORCE_AT LASTRETRY tells LOADPLUS to start canceling threads at the beginning of the last retry attempt following a drain process timeout. You can control the number of drain retry attempts with the RETRY command option or DRNRETRY installation option.

**FORCE_DELAY**

The FORCE_DELAY option tells LOADPLUS how long to wait, in hundredths of a second, before starting the thread cancelation process. For example, specify 7 to tell LOADPLUS to wait .07 seconds.

Specify an integer value of 0 or greater. A value of 0 tells LOADPLUS to start the cancelation process immediately upon reaching the point specified by the FORCE_AT option.

**Specifying the default**

You can specify a default for the FORCE_DELAY command option in your installation options module by using the second parameter of the FORCE_AT installation option. LOADPLUS was shipped with a default value of 3 for this parameter. The FORCE_DELAY command option overrides the default that is in the installation options module.
FORCE_RPT

The FORCE_RPT option tells LOADPLUS whether to display a report of the canceled threads.

Specifying the default

You can specify a default for the FORCE_RPT command option in your installation options module by using the FORCE_RPT installation option. LOADPLUS was shipped with a default value of NONE for this option. The FORCE_RPT command option overrides the default that is in the installation options module.

NO

NO tells LOADPLUS not to display the report.

YES

YES tells LOADPLUS to display the report. If your JCL includes a BMCFORCE DD statement, LOADPLUS sends the canceled threads report to that data set. Otherwise, the report is displayed in the LOADPLUS SYSPRINT.

ALL

FORCE ALL tells LOADPLUS to cancel both read and write claimers at the point specified by the FORCE_AT option.

FORCE_AT

For information about this option, see the FORCE_AT description under READERS in this topic.

FORCE_DELAY

For information about this option, see the FORCE_DELAY description under READERS in this topic.

FORCE_RPT

For information about this option, see the FORCE_RPT description under READERS in this topic.

REPORTONLY

FORCE REPORTONLY tells LOADPLUS to display a report of the threads that it would have canceled if you had specified FORCE ALL. When reporting which
threads it would have canceled, LOADPLUS takes into account the current values of FORCE_AT and FORCE_DELAY.

If your JCL includes a BMCFORCE DD statement, LOADPLUS sends the canceled threads report to that data set. Otherwise, the report is displayed in the LOADPLUS SYSPRINT.

**FORCE_AT**

For information about this option, see the FORCE_AT description under READERS in this topic.

**FORCE_DELAY**

For information about this option, see the FORCE_DELAY description under READERS in this topic.

**FORCE_RPT**

LOADPLUS ignores FORCE_RPT when you specify FORCE REPORTONLY.

**DSNUTILB**

The DSNUTILB option tells LOADPLUS whether to invoke DSNUTILB to pass processing to the IBM DB2 LOAD utility.

```
DSNUTILB YES
DSNUTILB NO
```

LOADPLUS uses this option to enable support for certain features. For the list of features, see “Load jobs that invoke DSNUTILB” on page 57.

**Specifying the default**

You can specify the default for the DSNUTILB command option in your installation options module by using the DSNUTILB installation option. LOADPLUS was shipped with a default value of YES for this option. The command option overrides the default that is in the installation options module.

**YES**

DSNUTILB YES tells LOADPLUS to invoke DSNUTILB when it is required to support the object type that is involved in the load. If you are loading a multi-table table space and any table in that table space (regardless of whether it is participating in the load) has a feature that is supported only by invoking DSNUTILB, LOADPLUS invokes DSNUTILB for the load job.
**Note**
LOADPLUS invokes DSNUTILB to enable new features quickly. LOADPLUS generally provides native support for these features (without invoking DSNUTILB) in a later version of the product or via PTF. You cannot use the DSNUTILB option (or any other option) to request that LOADPLUS invoke DSNUTILB for a feature that LOADPLUS supports natively.

For requirements and restrictions when LOADPLUS invokes DSNUTILB, see “Load jobs that invoke DSNUTILB” on page 57.

**NO**

DSNUTILB NO tells LOADPLUS not to invoke DSNUTILB when DSNUTILB is required to support the object type that is involved in the load. If DSNUTILB NO is in effect when it is required, LOADPLUS terminates.

**MAPDDN**

For the mapping data set for a DSNUTILB load, MAPDDN allows you to override the default &DDNAME variable for the data set name pattern. The MAPDDN option is applicable only for a DSNUTILB load.

![Diagram](image)

LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

**Specifying the default**

You can specify the default for the MAPDDN command option in your installation options module by using the MAPDDN installation option. LOADPLUS was shipped with a default value of SYSMAP for this option. The command option overrides the default that is in the installation options module.

**DECFLOAT_ROUNDMODE**

The DECFLOAT_ROUNDMODE option tells LOADPLUS how to round values in any DECFLOAT columns that are participating in the load.
If you do not specify a value, the utility uses the DECFloat round mode value from your DSNHDECP settings.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility.

**ROUND_CEILING**

ROUND_CEILING rounds toward positive infinity. If the extra digits are all zero or the number is negative, the utility truncates the digits. Otherwise, the utility rounds the remaining value up (increments the rightmost digit by 1).

**ROUND_DOWN**

ROUND_DOWN truncates the extra digits (rounds toward 0).

**ROUND_FLOOR**

ROUND_FLOOR rounds toward negative infinity. If the extra digits are all zero or the number is positive, the utility truncates the digits. Otherwise, the remaining value is negative and the utility rounds up (increments the rightmost digit by 1).

**ROUND_HALF_DOWN**

ROUND_HALF_DOWN rounds to the nearest number as explained in the following table:

<table>
<thead>
<tr>
<th>Extra digits</th>
<th>Utility action</th>
</tr>
</thead>
<tbody>
<tr>
<td>.5 or less</td>
<td>Truncates the digits (rounds the remaining value down)</td>
</tr>
<tr>
<td>Extra digits</td>
<td>Utility action</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td>Greater than .5</td>
<td>Rounds the remaining value up (increments the rightmost digit by 1)</td>
</tr>
</tbody>
</table>

**ROUND_HALF_EVEN**

ROUND_HALF_EVEN rounds to the nearest number as explained in the following table:

**Table 50: ROUND_HALF_EVEN rounding**

<table>
<thead>
<tr>
<th>Extra digits</th>
<th>Rightmost digit of remaining value</th>
<th>Utility action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Greater than .5</td>
<td>Not applicable</td>
<td>Rounds the remaining value up (increments the rightmost digit by 1)</td>
</tr>
<tr>
<td>Less than .5</td>
<td>Not applicable</td>
<td>Truncates the digits (rounds the remaining value down)</td>
</tr>
<tr>
<td>Equal to .5</td>
<td>Odd</td>
<td>Rounds the remaining value up (increments the rightmost digit by 1)</td>
</tr>
<tr>
<td>Equal to .5</td>
<td>Even</td>
<td>Truncates the digits (rounds the remaining value down)</td>
</tr>
</tbody>
</table>

**ROUND_HALF_UP**

ROUND_HALF_UP rounds to the nearest number as explained in the following table:

**Table 51: ROUND_HALF_UP rounding**

<table>
<thead>
<tr>
<th>Extra digits</th>
<th>Utility action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Less than .5</td>
<td>Truncates the digits (rounds the remaining value down)</td>
</tr>
<tr>
<td>.5 or greater</td>
<td>Rounds the remaining value up (increments the rightmost digit by 1)</td>
</tr>
</tbody>
</table>

**ROUND_UP**

ROUND_UP rounds away from 0. If the extra digits are all zero, the utility truncates the digits. Otherwise, the utility rounds the remaining value up (increments the rightmost digit by 1).

**LOADLOBDATA**

The LOADLOBDATA option tells LOADPLUS to invoke DSNUTILB if required to load LOB data in a LOB table space.
In most cases, LOADPLUS loads LOB data without invoking DSNUTILB. However, LOADPLUS invokes DSNUTILB when any of the following conditions exists:

- You are loading a partition-by-growth table space.
- You are running an SQLAPPLY load.
- You are running a two-phase load (PRELOAD CONTINUE).
- You are loading from a referenced file of a different type than the column type (for example, from a BLOB referenced file to a CLOB column).

Specify YES to tell LOADPLUS to load the corresponding LOB table space in these cases. The DSNUTILB installation options must also be set to YES:

**LOBAVGPCT**

The LOBAVGCPCT option provides LOADPLUS with an estimate of the size of your LOB data.

LOADPLUS uses this information to balance the number of tasks needed to load the base table, load the LOB table space, and build the indexes. LOADPLUS also uses this information when allocating LOB copy data sets.

Specify a percentage of the maximum size defined for the LOB that you are loading. Valid values are 1 through 100.

Alternatively, you can specify the AVGSIZE option on the field specification for the particular field that you are loading. The AVGSIZE field specification overrides the LOBAVGCPCT option. For more information, see “AVGSIZE” on page 353.

**Specifying the default**

You can specify the default for the LOBAVGCPCT command option in your installation options module by using the LOBAVGCPCT installation option.
LOADPLUS was shipped with a default value of 50 for this option. The command option overrides the default that is in the installation options module.

**XMLAVGSIZE**

The XMLAVGSIZE option provides LOADPLUS with an estimate of the size of your XML data.

```
XMLAVGSIZE 10 M K G
```

LOADPLUS uses this information to balance the number of tasks needed to load the base table, load the XML table space, and build the indexes. LOADPLUS also uses this information when allocating XML copy data sets.

Specify a number for the estimated size of your XML data followed by a unit of measure. The minimum value that you can specify is 1K. The maximum value that you can specify is 2G, 2048M, or 2097152K.

Alternatively, you can specify the AVGSIZE option on the field specification for the particular field that you are loading. The AVGSIZE field specification overrides the XMLAVGSIZE option. For more information, see “AVGSIZE” on page 355.

**Specifying the default**

You can specify the default for the XMLAVGSIZE command option in your installation options module by using the XMLAVGSIZE installation option. LOADPLUS was shipped with a default value of 10M for this option. The command option overrides the default that is in the installation options module.

**Statistics options**

The statistics options allow you to update statistical information in the DASD MANAGER PLUS database statistics tables and in the DB2 catalog.

LOADPLUS and REORG PLUS use the BMC Common Statistics component to update these statistics.

---

**Note**

The utility automatically updates the DB2 real-time statistics tables.
Reporting options

The Common Statistics component generates a report of the statistics that it updates.

You can direct the output of this report in any of the following ways:

- Send the output to a data set that is separate from the utility SYSPRINT.
  To use this method, specify an ASUSRPRT DD statement in your JCL.

- Send the output to the utility SYSPRINT.
  To use this method, do not specify an ASUSRPRT DD statement in your JCL. The utility automatically sends the report to your SYSPRINT.

  **Note**
  The Common Statistics component does not send this output to a SYSPRINT2 data set.

- Suppress the output.
  To suppress the output, specify `/ASUSRPRT DD DUMMY` in your JCL.

For descriptions of the fields that appear in this report, see the *DASD MANAGER PLUS for DB2 Reference Manual*.

**BMCSTATS**

The BMCSTATS option tells LOADPLUS whether to use the BMC statistics that it gathers to update the DASD MANAGER PLUS database statistics tables.

LOADPLUS ignores this option for either of the following types of load jobs:

- When invoking DSNUTILB
- For an SQLAPPLY load

**NO**

BMCSTATS NO, which is the default, tells LOADPLUS not to update the DASD MANAGER PLUS database statistics tables.
YES

BMCSTATS YES tells LOADPLUS to update the DASD MANAGER PLUS database statistics tables. This option requires that you have the DASD MANAGER PLUS product installed.

Restrictions

The following restrictions apply to BMCSTATS YES:

- LOADPLUS changes BMCSTATS YES to BMCSTATS NO when any of the following conditions exists:
  - You are loading a clone table.
  - You specify INDEX UPDATE.
  - You specify LOAD RESUME YES (unless you specify LOAD RESUME YES INDEX BUILD PART n REPLACE)
  - You restarted the job and, in the original job, a participating table space partition was completely loaded or an index partition was completely updated or built.
  - Statistics collection fails in a load or build task.

- The following discards are not reflected in the statistics:
  - For a single-phase load, LOB or XML discards and discards due to duplicates in unique non-data-sorting indexes
  - For a two-phase load, any discards that occur after the read tasks complete

Additional considerations

Note the following additional information for BMCSTATS YES:

- You must include the BMCPDWD data set in the STEPLIB of your load job.

- For any columns that have a FIELDPROC defined, LOADPLUS gathers statistics for the encoded values from the FIELDPROC. LOADPLUS does not gather statistics for column values that are stored in SYSIBM.SYSFIELDS.

- For multi-column keys, LOADPLUS uses only the first column to update statistics in the DASD MANAGER PLUS tables.

- You can control the amount of table space sampling that LOADPLUS does for these statistics. For more information, see “TSSAMPLEPCT” on page 293.
Due to differences in execution timing, options, and sampling, the statistics that LOADPLUS generates might differ slightly from the statistics produced by the BMCSTATS utility of DASD MANAGER PLUS.

REPORT

BMCSTATS REPORT tells LOADPLUS to gather statistics and produce the statistics report without updating the DASD MANAGER PLUS statistics tables.

Note

This option uses the same amount of processing overhead as specifying BMCSTATS YES.

Restrictions

The same restrictions that apply to BMCSTATS YES also apply to BMCSTATS REPORT. For information, see “YES” on page 290.

Additional considerations

Note the following additional information for BMCSTATS REPORT:

- For any columns that have a FIELDPROC defined, LOADPLUS gathers statistics for the encoded values from the FIELDPROC. LOADPLUS does not gather statistics for column values that are stored in SYSIBM.SYSFIELDS.

- For multi-column keys, LOADPLUS reports statistics based only on the first column.

- You can control the amount of table space sampling that LOADPLUS does for these statistics. For more information, see “TSSAMPLEPCT” on page 293.

- Due to differences in execution timing, options, and sampling, the statistics that LOADPLUS generates might differ slightly from the statistics produced by the BMCSTATS utility of DASD MANAGER PLUS.

UPDATEDB2STATS

UPDATEDB2STATS tells LOADPLUS whether to update statistics in the DB2 catalog.

- UPDATEDB2STATS
  - NO
  - YES
DB2 uses these statistics to determine the access paths that the DB2 optimizer selects.

**NO**

UPDATEDB2STATS NO, which is the default, tells LOADPLUS not to update statistics in the DB2 catalog.

**YES**

UPDATEDB2STATS YES tells LOADPLUS to update statistics in the DB2 catalog. LOADPLUS uses the BMC Common Statistics component to update access path statistics.

**Note**

LOADPLUS passes the following options to the Common Statistics component. LOADPLUS does not enable you to change these options.

- UPDATEDB2 ALL
- HISTORY ALL
- FORCEROLLUP N
- KEYCARD Y

For information about how these options affect which statistics are updated in the DB2 catalog, see the DASD MANAGER PLUS documentation.

**Restrictions**

LOADPLUS does not update statistics in the DB2 catalog when any of the following conditions exists. If you encounter these circumstances, BMC recommends that you run the RUNSTATS utility to update statistics in the DB2 catalog.

- LOADPLUS changes UPDATEDB2STATS YES to UPDATEDB2STATS NO when any of the following conditions exists:
  - You are loading a clone table.
  - You specify INDEX UPDATE.
  - You specify LOAD RESUME YES (unless you specify LOAD RESUME YES INDEX BUILD PART \textit{n} REPLACE).
  - You restarted the job and, in the original job, a participating table space partition was completely loaded or an index partition was completely updated or built.
  - Statistics collection fails in a load or build task.
- The following discards are not reflected in the statistics:
— For a single-phase load, LOB or XML discards and discards due to duplicates in unique non-data-sorting indexes

— For a two-phase load, any discards that occur after the read tasks complete

Additional considerations

The following additional considerations apply to UPDATEDB2STATS YES:

■ When invoking DSNUTILB, LOADPLUS passes this option to the STATISTICS option of the IBM DB2 LOAD utility only if LOAD RESUME NO SHRLEVEL NONE is in effect.

LOADPLUS passes UPDATEDB2STATS YES to the DB2 LOAD utility as STATISTICS TABLE (ALL) INDEX (ALL) UPDATE ALL HISTORY ALL FORCEROLLUP YES.

■ For multi-column keys, LOADPLUS uses only the first column to update statistics in the SYSIBM.SYSCOLUMNS table. If the cardinality for the first key column changes dramatically, BMC recommends that you run the RUNSTATS utility to ensure that the DB2 optimizer selects the appropriate path.

■ You can control the amount of table space sampling that LOADPLUS does for these statistics. For more information, see “TSSAMPLEPCT” on page 293.

TSSAMPLEPCT

The TSSAMPLEPCT option enables you to specify a percentage of table space pages that you want LOADPLUS to sample when gathering statistics.

The following values are valid:

■ 1 through 50 tells LOADPLUS to sample the specified percentage of the table space pages.

■ 100 tells LOADPLUS to read all table space pages instead of sampling.

**Note**

Values 51 through 99 are not valid.

Restrictions
LOADPLUS ignores the TSSAMPLEPCT option for either of the following types of load jobs:

- A DSNUTILB load
- An SQLAPPLY load

**Specifying the default**

You can specify the default for the TSSAMPLEPCT command option in your installation options module by using the TSSAMPLEPCT installation option. LOADPLUS was shipped with a default value of 100 for this option. The command option overrides the default that is in the installation options module.

**Copy options**

The copy options indicate whether LOADPLUS should create a copy of the table space, and, if so, what type of copy.

Additional options control other aspects of the copy process, including the ability to override the default ddnames of the data sets.

**COPY**

The COPY option tells LOADPLUS whether to produce a copy of the table space or selected partitions during the load.

Additional options tell LOADPLUS what type of copy to produce.

In addition to interactions with other options (which are documented with those options), the value that you specify can affect the following aspects of your load job:
Required data sets (“Copy data sets in LOADPLUS” on page 436)

Restart (“Copy data sets” on page 469 and “Inline image copies” on page 474)

**NO**

COPY NO (the default) tells LOADPLUS to not make a copy of the table space.

**COPYPEND**

The COPYPEND option tells LOADPLUS whether to set the COPY pending status when it finishes the load.

**Specifying the default**

You can specify the default for the COPYPEND command option in your installation options module by using the COPYPEND installation option. LOADPLUS was shipped with a default value of YES for this option. With the following exception, the command option overrides the default that is in the installation options module.

If the value of the COPYPEND installation option includes the ENFORCE keyword, you cannot override the installation option with the COPYPEND command option.

**YES**

YES tells LOADPLUS to set the COPY pending status after loading the table. The job completes with return code 4, indicating that you need to make a copy of the table for recovery purposes.

**Restrictions**

Note the following restrictions when specifying COPYPEND YES:

- For an SQLAPPLY load, LOADPLUS ignores this option.

- When you specify LOAD RESUME YES and you do not load any rows, LOADPLUS does not set COPY pending and the job completes with return code 0.

- If the table space that you are loading is in LPL or WEPR status and you specify this option, LOADPLUS terminates.

- If the value for the COPYPEND installation option is (NO,ENFORCE), specifying COPYPEND YES on the LOAD command causes LOADPLUS to issue message BMC50115E and terminate.
NO

NO tells LOADPLUS not to set the COPY pending status after loading the table. The job completes with return code 0. Use this option only for static tables or tables that you can reload if needed, because you will not be able to recover the tables.

For a DSNUTILB load, LOADPLUS passes COPYPEND NO as NOCOPYPEND to the IBM DB2 LOAD utility.

Restriction

The following restrictions apply to COPYPEND NO:

- If the value for the COPYPEND installation option is (YES,ENFORCE), specifying COPYPEND NO on the LOAD command causes LOADPLUS to issue message BMC50115E and terminate processing.

- When you specify COPYPEND NO, LOADPLUS does not reset (remove) COPY pending status if the table that you are loading was in COPY pending status before the load.

YES

If you specify COPY YES, LOADPLUS creates a copy of the table space or partitions, depending on the type of object that you are loading and the value of your other command or installation options. Depending on the value of other copy options, LOADPLUS creates a standard DB2 image copy, DSN1COPY-type copy, or inline image copy.

Restriction

If you specify this option for an SQLAPPLY load, LOADPLUS issues message BMC50115E and terminates.

DSNUTILB load jobs

The following considerations apply when you specify COPY YES for a DSNUTILB load:

- When invoking DSNUTILB, LOADPLUS includes the COPYDDN and RECOVERYDDN information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

- You must specify ACTIVE YES for at least the primary copy data set (DDTYPE LOCPFCPY). Otherwise, LOADPLUS issues message BMC50181E and terminates. All copy data sets for a DSNUTILB load are dynamically allocated. LOADPLUS ignores any copy data set allocation in your JCL.
Additional considerations

The following additional considerations apply to COPY YES:

- If you specify this option and you are loading LOB or XML data, you must enable dynamic allocation for the appropriate LOB or XML copy data sets.

- If the table space was in COPY pending status before the load, LOADPLUS resets COPY pending status unless you specify REGISTER NONE.

- If you specify this option and the table space that you are loading is in LPL or WEPR status, you must register at least one copy.

- LOADPLUS creates all copies with system pages at the beginning of the data set (in the same way that the IBM COPY utility creates image copies when you specify SYSTEMPAGES YES).

- The difference between an image copy and a DSN1COPY-type copy is one of registration: an image copy is registered, a DSN1COPY-type copy is not.

INLINE

The INLINE option tells LOADPLUS whether to make an inline image copy while loading the table space.

Restriction

When invoking DSNUTILB, LOADPLUS ignores this option.

Specifying the default

You can specify the default for the INLINE command option in your installation options module by using the INLINECP installation option. LOADPLUS was shipped with a default value of YES for this option. The command option overrides the default that is in the installation options module.

YES

INLINE YES tells LOADPLUS to create an inline image copy while loading the table space. Because LOADPLUS creates the image copy while loading the table space, rather than after, specifying INLINE YES can reduce the elapsed time of your load job.

Restrictions

When any of the following conditions exists, LOADPLUS functions as if you specified INLINE NO, regardless of the option that you specify here or in your installation options:
You are loading XML data.

You specify LOAD RESUME YES, unless you also specify PART n REPLACE for all participating partitions.

The copy data sets are on a stacked tape.

You restart your load job.

An exception to this condition is that, if you are running a two-phase load job and you specify RESTART(PHASE), LOADPLUS generates an inline copy if the value of INLINE is YES.

The size of the table space page is greater than 4 KB, you are loading multiple partitions, and you have one image copy data set.

An exception to this condition exists for partition-by-growth table spaces. LOADPLUS can create inline image copies of partition-by-growth table spaces in this case.

NO

INLINE NO tells LOADPLUS not to create an inline image copy while loading the table space, but to create a DB2 image copy or DSN1COPY-type copy after loading the table space.

REGISTER

The REGISTER option tells LOADPLUS to register some, none, or all copies with DB2 in the SYSIBM.SYSCOPY table. You can specify one of the following options:

**Table 52: REGISTER option values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
<td>ALL, which is the default, registers all copies requested in the JCL by ddname.</td>
</tr>
<tr>
<td>NONE</td>
<td>NONE tells LOADPLUS to not register any of the copies with DB2. Unless the object was in COPY pending status before the load, LOADPLUS does not leave the object in COPY pending status. If you specify this option and the table space that you are loading is in LPL or WEPR status, LOADPLUS terminates.</td>
</tr>
<tr>
<td>ddname</td>
<td>This option allows you to specify by ddname the copy data set to register. LOADPLUS registers only the copies in the data sets that you specify with this option. The ddnames that you specify in this option must also be present in your JCL.</td>
</tr>
</tbody>
</table>

**Restrictions**

Note the following restrictions on registering copies:
When invoking DSNUTILB, LOADPLUS ignores this option.

LOADPLUS does not register the following copies:
- A copy written to DD DUMMY
- A copy written to a DD specified with DSN=NULLFILE

If any one copy is rejected due to a duplicate entry in the SYSIBM.SYSCOPY table, LOADPLUS does not register any of the requested copies. LOADPLUS issues an error message and completes with return code 8.

**COPYLVL**

COPYLVL tells LOADPLUS how to assign the dynamically allocated image copy data sets in certain cases. LOADPLUS uses this option only when dynamic allocation is active for copy data sets.

The COPYLVL option applies to dynamically allocated copy data sets when either of the following conditions exist:

- You are loading all partitions of a partitioned table space.
- You are loading a subset of contiguous partitions by using the PART command option, and you specify YES for the COPYSUBSET installation option.

You can improve performance when loading partitioned table spaces by specifying COPYLVL PART.

**Specifying the default**

You can specify the default for the COPYLVL command option in your installation options module by using the COPYLVL installation option. LOADPLUS was shipped with a default value of FULL for this option. The command option overrides the default that is in the installation options module.

**FULL**

COPYLVL FULL tells LOADPLUS to allocate a single copy data set to contain all of the partitions being loaded.

**Restrictions**
LOADPLUS changes COPYLVL FULL to COPYLVL PART when any of the following conditions exists:

- You specify a subset of partitions, but the value of the COPYSUBSET installation option is NO.

- The value of the COPYSUBSET installation option is YES and you specify a subset of partitions, but they are not contiguous.

- For XML copies, you specify LOAD RESUME YES (without PART REPLACE) COPY YES for a partition-by-growth XML table space.

  **Note**
  For base table space copies, LOADPLUS honors COPYLVL FULL.

**PART**

COPYLVL PART tells LOADPLUS to allocate individual copy data sets for each partition being loaded.

**Restriction**

This option is not valid when LOADPLUS invokes DSNUTILB (and you specify COPY YES). LOADPLUS issues message BMC50178E and terminates.

**Additional considerations**

Note the following considerations for this option:

- If you are using a GDG name, each partition must have a different GDG base.

- If you are loading more than 99 partitions, use the COPYDDN option to specify a ddname prefix for the copy data sets that results in eight characters or less after LOADPLUS appends the highest partition number. If you are also making remote copies, use the RECOVERYDDN option in the same way.

- If you specify a tape device for the UNIT option, LOADPLUS allocates a tape unit for each partition.

**COPYDDN**

COPYDDN allows you to override the default ddnames or ddname prefixes of your local copy data sets that were specified in your installation options module.
The ddnames correspond to the data sets that receive either an image copy or DSN1COPY-type copy of the table space or partitions that you are loading.

If you specify ddname2 for this option, you must either allocate this data set in your JCL or dynamic allocation must be active for both the primary and backup copy data sets. If you specify ddname2 and dynamic allocation is active for only the primary copy data set, LOADPLUS terminates because it expects a second copy data set.

If you are registering the copies, ddname1 will be the DB2 local primary and ddname2 will be the local backup. For information about using these data sets in LOADPLUS, see “Copy data sets in LOADPLUS” on page 436.

If you use this option to override the default name in the installation options module, you must also change the name in your JCL.

When LOADPLUS invokes DSNUTILB (and you specify COPY YES), LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

**Restriction**

This option does not apply to LOB or XML copy data sets.

**Partition-level copies**

If you are making partition-level copies, specify only the ddname prefix (no nn) in this option. If you are dynamically allocating copy data sets (and you specify COPIYLV PART), LOADPLUS appends the partition number to the ddname prefix. Specify a prefix that results in eight characters or less after LOADPLUS appends the highest partition number. For more information, see “Specifying ddname prefixes” on page 81.

If you are not dynamically allocating copy data sets, the ddname that you specify in the JCL must have the partition number nn appended to this prefix. The length of nn can be from one through seven characters, depending on the length of the prefix. The length of the ddname with the prefix must be eight characters or less. For example, if you are loading partition 15, you could specify ddname1 as BCOPY, and specify BCOPY015 in your JCL.
If you are loading a partition-by-growth table space, ensure that the ddname prefix that you specify allows for the number of partitions specified for MAXPARTITIONS in your table space definition.

**Dynamic allocation**

If dynamic allocation is active and you specify more than one ddname prefix for dynamic allocation, the prefix for each ddname must be different enough for LOADPLUS to differentiate one prefix from another. To be different enough, if these prefixes are different only because one prefix has additional trailing bytes, then these trailing bytes must contain at least one nonnumeric byte. For example, the first set of prefixes that follow is different enough, but the second set is not:

- **Acceptable set:**
  
  | BMCRD  |
  | BMCRDwk |

- **Unacceptable set:**
  
  | BMCRD  |
  | BMCRD11 |

**Specifying the default**

You can specify the default for the COPYDDN command option in your installation options module by using the COPYDDN installation option. LOADPLUS was shipped with a default value of (BMCCPY,BMCCPZ) for this option. The command option overrides the default that is in the installation options module.

**RECOVERYDDN**

RECOVERYDDN allows you to override the default ddnames or ddname prefixes of your remote copy data sets that were specified in your installation options module.

```
( BMCRCY
  ddbname1

, BMCRCZ
  ddbname2
)
```

The ddnames correspond to the data sets that receive either an image copy or DSN1COPY-type copy of the table space or partitions that you are loading.
If you specify \textit{ddname2} for this option, you must either allocate this data set in your JCL or dynamic allocation must be active for both the primary and backup remote copy data sets. If you specify \textit{ddname2} and dynamic allocation is active for only the primary copy data set, LOADPLUS terminates because it expects a second copy data set.

If you are registering the copies, \textit{ddname1} will be the DB2 remote primary and \textit{ddname2} will be the remote backup. For information about using these data sets in LOADPLUS, see “Copy data sets in LOADPLUS” on page 436.

If you use this option to override the default name in the installation options module, you must also change the name in your JCL.

When LOADPLUS invokes DSNUTILB (and you specify COPY YES), LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

\textbf{Restriction}

This option does not apply to LOB or XML copy data sets.

\textbf{Partition-level copies}

If you are making partition-level copies, specify only the ddname prefix (no \textit{nn}) in this option. If you are dynamically allocating copy data sets (and you specify COPYLVL PART), LOADPLUS appends the partition number to the ddname prefix. Specify a prefix that results in eight characters or less after LOADPLUS appends the highest partition number. For more information, see “Specifying ddname prefixes” on page 81.

If you are not dynamically allocating copy data sets, the ddname that you specify in the JCL must have the partition number \textit{nn} appended to this prefix. The length of \textit{nn} can be from one through seven characters, depending on the prefix length. The maximum length of the ddname with the prefix must be eight bytes or less. For example, if you are loading partition 203, you could specify \textit{ddname1} as BMCRY, and specify BMCRY203 in your JCL.

If you are loading a partition-by-growth table space, ensure that the ddname prefix that you specify allows for the number of partitions specified for MAXPARTITIONS in your table space definition.

\textbf{Dynamic allocation}

If dynamic allocation is active and you specify more than one ddname prefix for dynamic allocation, the prefix for each ddname must be different enough for LOADPLUS to differentiate one prefix from another. To be different enough, if these prefixes are different only because one prefix has additional trailing bytes, then these trailing bytes must contain at least one nonnumeric byte. For example, the first set of prefixes that follow is different enough, but the second set is not:
Specifying the default

You can specify the default for the RECOVERYDDN command option in your installation options module by using the RCVYDDN installation option. LOADPLUS was shipped with a default value of (BMCRCY,BMCRCZ) for this option. The command option overrides the default that is in the installation options module.

INTO TABLE options

The options that you specify with INTO TABLE describe the objects and data that you want to load.

INTO TABLE block

This block contains the keywords that make up the INTO TABLE option.

For complete details, see "INTO TABLE" on page 306

SKIPFIELDS

This option tells LOADPLUS whether to allow fields in your field specifications whose names do not match any column name of any table that is specified in an INTO statement on a single LOAD command.

These "skipped" fields are used as placeholders in the input record or are used in predicates, such as NULLIF, for conditional processing.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility as IGNOREFIELDS.
YES

SKIPFIELDS YES, which is the default, specifies that LOADPLUS not diagnose any CHARACTER, VARCHAR, VARGRAPHIC, VARBINARY, or ROWID field whose name does not match any column name of a table that is specified in an INTO statement.

Depending on column definitions, LOADPLUS loads nulls, specified default values, or generated values into the columns that are associated with the skipped fields.

ALLTYPES

ALLTYPES specifies that LOADPLUS not diagnose any field, regardless of type, whose name does not match any column name of a table that is specified in an INTO statement.

NO

SKIPFIELDS NO tells LOADPLUS to fail if any field name that is specified in an INTO statement does not match a column name of the table that you are loading. Specifying this option prevents errors in which misspelled field names in the field specification cause LOADPLUS to load nulls (if the column is nullable) or default values (if the column is defined as NOT NULL WITH DEFAULT).

UNIQUEINTO

The UNIQUEINTO option specifies whether each record matches only one INTO specification when you include multiple INTO statements on your LOAD command.

For information about how to enhance LOADPLUS performance when using this option, see “Performance considerations for the INTO and UNIQUEINTO command options” on page 660.

NO

UNIQUEINTO NO, which is the default, tells LOADPLUS to try to match each input record to each INTO specification.

Restriction
When invoking DSNUTILB, LOADPLUS ignores this option.

**YES**

UNIQUEINTO YES tells LOADPLUS that, once an INTO specification selects an input record, LOADPLUS should not test that record against other INTO specifications.

Using UNIQUEINTO YES where appropriate can dramatically reduce execution times for multi-partition loads and multi-table loads if each input record matches at most one INTO specification.

**Restriction**

This option is not valid when LOADPLUS invokes DSNUTILB. LOADPLUS issues message BMC50178E and terminates.

**INTO TABLE**

The INTO TABLE option identifies a table to be loaded.

If you do not qualify the table name with the creator name, LOADPLUS uses the creator name that was established when LOADPLUS connected to DB2.
Requirements

Note the following requirements when specifying INTO TABLE:

■ LOADPLUS does not support Unicode table names.

■ You cannot specify a view or an alias.

■ If the table uses a VALIDPROC, an EDITPROC, or both, you must include the routines in your LINKLIST, JOBLIB, or STEPLIB.

■ You must specify at least one INTO TABLE option for each table that you load. All tables that you specify must reside in the same table space.

■ If you want LOADPLUS to check referential constraints, you cannot specify both the parent and child tables in your INTO statements in a single load execution.

■ For a DSNUTILB load, if you are loading a compressed table that contains more than 255 partitions, you must specify a separate INTO statement for each partition that you are loading.

Additional guidelines

Use the following guidelines to determine how many INTO statements to use for your job and when to use the PART option:

■ If you are replacing data in multiple tables in a segmented table space, specify a separate INTO statement for each table as shown in the following example:

```
LOAD RESUME YES
  INTO TABLE tableName1 REPLACE
  INTO TABLE tableName2 REPLACE
```

■ Specify a separate INTO statement for each table in a segmented table space when the following conditions exist:
  — You specify LOAD RESUME YES INTO TABLE tableName REPLACE.
  — The table space is in REORP status.
  — You want LOADPLUS to reset REORP status.

■ For optimal performance when either of the following conditions exists, specify only one INTO statement without the PART option. The resulting partitioning key determines the partition into which the record is loaded, so the PART option is not necessary.
  — You specify LOAD REPLACE
  — You specify LOAD RESUME YES and are loading the majority of partitions in a partitioned table space

■ If you specify LOAD RESUME YES and are loading only a few partitions, specify all partitions on one INTO statement (for example, INTO PART 1:3,5). This
specification eliminates unloading and loading all partitions of the partitioned index and can save considerable time.

- With the exception described in the requirements, avoid specifying a separate INTO statement with PART option for each partition.

**Additional considerations**

Note the following additional information:

- When invoking DSNUTILB, LOADPLUS typically passes the INTO statement as is to the IBM DB2 LOAD utility, with the following exceptions:
  
  — LOADPLUS translates the TRIM option on your field specification to the appropriate STRIP specification.

  — LOADPLUS does not pass the INTO statement to the DB2 LOAD utility when you include either of the following specifications. In these cases, LOADPLUS issues message BMC50178E and terminates.
    
    — INTO TABLE `tableName` REPLACE
    — POS instead of POSITION on the field specification

  With the exception of TRIM, the syntax that you specify on the INTO statement must be valid for both LOADPLUS and the DB2 LOAD utility.

- Unless you specify UNIQUEINTO, each input file record is evaluated for each INTO statement.

- A record might not be loaded when the following conditions exist:
  
  — You specify multiple INTO statements that LOADPLUS evaluates for the same input file record.

  — That record is valid for one INTO statement but not another.
    
    In this case, the invalid input record is not loaded.

- For DB2 subsystems that are defined as MIXED=YES, LOADPLUS supports the use of DBCS characters. You can use DBCS characters in DB2 identifiers, such as table or column names, or in constants and field names specified in your INTO TABLE option.

- For information about how to enhance LOADPLUS performance when using this option, see “Performance considerations for the INTO and UNIQUEINTO command options” on page 660.
PERIODOVERRIDE

The PERIODOVERRIDE option tells LOADPLUS to load values from an input file into a column that is participating in the PERIOD constraint of a temporal table and is defined with GENERATED ALWAYS. If you specify both PERIODOVERRIDE and TRANSIDOVERRIDE, you must specify PERIODOVERRIDE before TRANSIDOVERRIDE.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility.

TRANSIDOVERRIDE

The TRANSIDOVERRIDE option tells LOADPLUS to load values from an input file into a transaction-start-ID column that is defined with GENERATED ALWAYS. If you specify both PERIODOVERRIDE and TRANSIDOVERRIDE, you must specify PERIODOVERRIDE before TRANSIDOVERRIDE.

When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility.

REPLACE

The REPLACE option tells LOADPLUS to replace the data in the specified table of a segmented table space. You can use this option for multiple tables in a segmented table space by specifying a separate INTO statement for each table. To replace data in all tables of a table space, BMC recommends that you specify LOAD REPLACE, as described in “REPLACE” on page 187.

Restrictions

Note the following restrictions when you specify TABLE tableName REPLACE:

- LOADPLUS terminates if you are loading a system-period temporal table that is enabled for versioning.

- If you are loading LOB or XML data, this option applies only to the base table. When you specify this option, LOADPLUS loads the LOB or XML table space as if you specified LOAD REPLACE.

- This option is not valid when LOADPLUS invokes DSNUTILB. LOADPLUS issues message BMC50178E and terminates.

- This option is valid only for segmented table spaces and universal table spaces. For any other type of table space, LOADPLUS terminates.
To avoid inefficient use of space, BMC recommends that you do not use TABLE <table_name> REPLACE to load a table in a universal table space.

- If you specify REPLACE for one table, you must specify REPLACE for all tables that are participating in the load. In other words, in the same load job you cannot replace data in some tables and add to existing data (resume) in other tables.

- LOADPLUS does not update indexes for this type of load. If you specify INDEX UPDATE, LOADPLUS changes this option to INDEX BUILD.

- LOADPLUS does not update the DB2 real-time statistics tables in this case.

**PART**

The PART option specifies the partition number of a partitioned table space that you want to load. LOADPLUS loads only data within the limit key range of the specified partition. You can specify one, some, or all of the partitions either individually or by specifying a range of partitions. If you do not specify PART, LOADPLUS loads all partitions of the table space.

For guidelines about when to use the PART option, see “INTO TABLE” on page 306.

**Specifying partition numbers**

Note the following information about how to specify partition numbers with the PART option:

- You can specify integer values from 1 through 4096.

- Individual partitions in a list can be in any order (but partitions within a range must be in ascending order).

- You can specify a mixture of individual partitions and ranges of partitions.

- If you specify a partition number more than once, LOADPLUS ignores any occurrence after the first.

- If you want to create a single image copy for a subset of partitions, specify only one set of contiguous partitions, either individually or as a range.

The following example illustrates a valid PART specification:

```
PART 10,1:5,7
```

**Additional considerations**
Note the following additional information about the PART option:

- The PART option is not valid for partition-by-growth table spaces.

- If you specify multiple partition numbers on one INTO TABLE block, LOADPLUS allows only one INTO TABLE block.

- LOADPLUS optimizes the processing of INTO PART statements by combining statements that are the same. For example, if you specify INTO PART 1 twice on the same LOAD command, LOADPLUS combines these statements into a single INTO statement and loads one set of rows. In contrast, DB2 loads two sets of rows into partition 1 in this example.

- If data exists in partitions that are not specified on an INTO TABLE option, specify RESUME YES to save that data. Otherwise, processing terminates.

- You cannot change partition numbers when restarting your load job.

- Loading a large number of compressed partitions might result in constrained resources. If you encounter this problem, consider specifying smaller groups of partitions in multiple load steps.

PREFORMAT

The PREFORMAT option tells LOADPLUS whether to preformat the unused pages of the data set. If you specify PREFORMAT without a value, LOADPLUS assumes PREFORMAT YES.

LOADPLUS also provides PREFORMAT support at the global level. See “PREFORMAT” on page 271.

Specifying the default

You can specify the default for the PREFORMAT option in your installation options module by using the PREFORMAT installation option. LOADPLUS was shipped with a default value of NO for this option. The command option overrides the default that is in the installation options module.

YES

PREFORMAT YES tells LOADPLUS to preformat the unused portion of the data set. Preformatting writes full pages that have been initialized with zeros up to the high-allocated RBA of the table space partition and index partition. Preformatting occurs after LOADPLUS loads the data and builds the indexes.

For an SQLAPPLY load, LOADPLUS ignores this option and issues message BMC50109I.
PREFORMAT NO tells LOADPLUS not to preformat the unused pages.

REPLACE

If you specify PART REPLACE, LOADPLUS deletes the existing data in the partition before it loads the data. To delete existing data in all partitions, including those that you do not specify on an INTO option, use LOAD REPLACE.

If you also specify INDEX UPDATE, LOADPLUS rebuilds the data-sorting indexes and updates the participating non-data-sorting indexes and nonpartitioned data-sorting indexes.

LOADPLUS converts data that is in basic row format to reordered row format, but only for the partitions that you are replacing. However, if the table contains a VALIDPROC or EDITPROC, LOADPLUS loads the data according to the existing row format on the table space.

Requirements and restrictions

If any of the partitions that you specify are in REORP status, you must specify all partitions that are in REORP status.

In addition to restrictions on the PART option itself, LOADPLUS terminates when you specify PART REPLACE and any of the following conditions exists:

- You are running an SQLAPPLY load.
- You are loading a table space that contains an XML column.
- You are loading a partition-by-growth table space.
- You are loading a system-period temporal table that is enabled for versioning.

KEEPDICTIONARY

This option tells LOADPLUS whether to keep the existing compression dictionary for the partition that you specify with the PART option. If you specify KEEPDICTIONARY without a value, LOADPLUS assumes KEEPDICTIONARY YES. For additional considerations when using compression, see “Considerations for table space compression” on page 140.

LOADPLUS also provides KEEPDICTIONARY support at the global level. See “KEEPDICTIONARY” on page 188 for information.

The following table describes the values that you can specify for the KEEPDICTIONARY option:
Table 53: Values for the partition-level KEEPDICTIONARY option

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>If you specify KEEPDICTIONARY YES, LOADPLUS keeps the existing compression dictionary. If a dictionary already exists, LOADPLUS uses it for compression. If a dictionary does not exist, LOADPLUS builds one in the PRELOAD phase or the COMBINED phase. After completely building the dictionary, LOADPLUS compresses the data.</td>
</tr>
<tr>
<td>NO</td>
<td>If you specify KEEPDICTIONARY NO, LOADPLUS builds a new compression dictionary. LOADPLUS builds the dictionary in the PRELOAD phase or the COMBINED phase. After completely building the dictionary, LOADPLUS compresses the data.</td>
</tr>
</tbody>
</table>

**Restrictions**

The following restrictions apply to the KEEPDICTIONARY option:

- This option is valid only if the table space partition that you are loading has the COMPRESS YES attribute.
- If a partition is compressed and a load job would convert the row format from BRF to RRF, LOADPLUS builds a new dictionary except when both of the following options are in effect:
  - The value of the LOADPLUS KEEPDICTIONARY option is YES.
  - The value of the DB2 subsystem parameter HONOR_KEEPDICTIONARY is YES.

**Specifying the default**

You can specify the default for the KEEPDICTIONARY option in your installation options module by using the KEEPDICTIONARY installation option. LOADPLUS was shipped with a default value of NO for this option. The command option overrides the default that is in the installation options module.

**WHEN**

The WHEN keyword is valid with INTO TABLE options.
To see where the WHEN syntax fits in the INTO TABLE option, see the diagram in “INTO TABLE” on page 306.

You can combine the WHEN keyword with a condition that allows you to specify the condition that must be true for LOADPLUS to select a row. You can also use the WHEN keyword with a TABLE=obid option for translating the input table OBID in the input record. This section describes the TABLE=obid specification first and follows with a complete discussion of conditions and predicates.

The value of the RULES installation option designated at installation affects the ways in which WHEN and data conversion, including NULLIF and DEFAULTIF processing, work:

- With RULES=STANDARD, LOADPLUS first compares the values in the input file to the WHEN condition. LOADPLUS then completes, on the resulting records, any data conversion and conversion due to NULLIF and DEFAULTIF processing.

- With RULES=BMC, LOADPLUS first completes any data conversion and conversion due to NULLIF and DEFAULTIF processing. LOADPLUS then compares values to the WHEN condition based on the type of comparison:
  - For predicates with column names on the left side, LOADPLUS compares the WHEN condition to the converted data.
  - For predicates with (start:end) or field names on the left side, LOADPLUS compares the WHEN condition to the original input data.
Therefore, the same LOAD command can produce different results, depending on which option was specified at LOADPLUS installation. For details and examples, see “RULES installation option” on page 819.

**TABLE=obid**

If you load data by using the FORMAT UNLOAD, FORMAT BMC, or FORMAT BMCUNLOAD option, you can specify the TABLE=obid option. LOADPLUS loads only those records that match the specified table OBID.

The following table describes considerations for using this option with each FORMAT option:

<table>
<thead>
<tr>
<th>FORMAT option</th>
<th>Considerations</th>
</tr>
</thead>
</table>
| FORMAT UNLOAD       | Specify TABLE=obid to translate the input table OBID in the input record when you are loading data from REORG UNLOAD ONLY output, or from a REORG PLUS SYSARC data set when ARCHFORMAT is DB2. Specify the OBID as a decimal number.  
  **Note:** One application for this option with FORMAT UNLOAD is to load data if the table OBID has changed. This situation might occur, for example, if you have performed the following steps:  
  1. Performed a REORG UNLOAD ONLY  
  2. Issued a DROP TABLE command  
  3. Issued an SQL CREATE TABLE statement (using the same column definitions)  
   To use this option under these circumstances, specify the table’s old OBID in TABLE=obid. |
| FORMAT BMC          | Specify TABLE=obid to translate the input table OBID in the input record when you are loading data from a REORG PLUS SYSARC data set. Specify the OBID as a decimal number.                                          |
| FORMAT BMCUNLOAD    | You must include TABLE=obid when you specify FORMAT BMCUNLOAD (which enables you to load data from UNLOAD PLUS). Specify the OBID as a decimal number.  
  In this case, the OBID is the object identifier that UNLOAD PLUS assigned when unloading the data, not the table’s DB2 object identifier. The LOAD control cards that UNLOAD PLUS generated contain this OBID. For an example, see “Example 14 LOAD REPLACE from UNLOAD PLUS data in internal format” on page 588. |
condition block

A condition is an expression that combines predicates. Predicates (see “predicates” on page 316) are comparisons of a field name, column name, or (start:end) field to a constant. You can use conditions and predicates with the WHEN option. (You can also use predicates with the NULLIF and DEFAULTIF options.)

A condition uses the parenthetical operators ( ) and the operators AND, OR, and NOT. The following table shows the results of the WHEN condition when you use the AND, OR, or NOT operator:

Table 55: Truth table for AND, OR, NOT

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>p AND q</th>
<th>p OR q</th>
<th>NOT p</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
</tbody>
</table>

LOADPLUS evaluates conditions and predicates left to right, with the following operator precedence:

- ( )
- NOT
- AND
- OR

As soon as LOADPLUS determines the resulting value of the overall condition, evaluation of any further predicates stops. If the result is true, the row is selected. If the result is false, the row is not selected.

predicates

Predicates are comparisons of the contents of a field, column, or (start:end) string to a constant, as shown in the following examples.
LOADPLUS uses predicates as part of WHEN, NULLIF, and DEFAULTIF syntax. Predicate operation differs based on the value of the RULES installation option.

Predicate examples

\[ \text{CITY='AUSTIN'} \]
\[ (1:6)\text{¬=}X'00000000000C' \]
\[ \text{INT\_COL > 5} \]

The following table defines how we use the terms `fieldName` and `columnName` when describing the left side of a predicate. The sections that follow this table describe these specifications in more detail.

**Table 56: Definition of `fieldName` and `columnName` for predicates**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>A name included in a field specification that does not correspond to a column in the table that you are loading</td>
</tr>
</tbody>
</table>
LOADPLUS command options

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnName</td>
<td>A column of the table that you are loading. This column must be included in a field specification unless you also specify FORMAT BMC, FORMAT UNLOAD, or FORMAT BMCUNLOAD.</td>
</tr>
</tbody>
</table>

**fieldName**

When you specify a name on your predicate and no column of the same name exists in the table that you are loading, LOADPLUS assumes that you have specified a field name. This field name must be included in your field specification and the length of the field must be less than or equal to 255.

**Restrictions**

Field names in predicates have the following restrictions:

- You cannot specify a row ID or identity column.

- You cannot specify a field name when you use WHEN with FORMAT BMC, FORMAT UNLOAD, or FORMAT BMCUNLOAD. You must specify a column name.

- For restrictions on the constants that you can specify with a field name in a predicate, see Table 58 on page 321.

**columnName**

If you specify a column name, it must be a column of the named table, and the length of the column must be less than or equal to 255. Column names must have corresponding field specifications unless you also specify FORMAT BMC, FORMAT UNLOAD, or FORMAT BMCLOAD. LOADPLUS accepts column names that include DBCS characters.

**Restrictions**

Column names in predicates have the following restrictions:

- You cannot specify a row ID or identity column.

- For restrictions on the constants that you can specify with a column name in a predicate, see Table 58 on page 321.

**(start:end)**

If you specify (start:end), the string defined by (start:end) must be contained entirely in the input record. The length indicated by (start:end) must not be greater than 255.
or less than the length of the string. If you do not specify the ending position, LOADPLUS uses the length of the constant for the comparison.

**Restrictions**

Note the following restrictions when using `(start:end)`:

- If you specify WHEN with FORMAT BMC, FORMAT UNLOAD, FORMAT BMCUNLOAD, or FORMAT CSV, you cannot specify `(start:end)`. For FORMAT BMC, FORMAT UNLOAD, and FORMAT BMCUNLOAD, you must specify a column name. For FORMAT CSV, you must specify a field name or column name.

- For restrictions on the constants that you can specify with `(start:end)` in a predicate, see Table 58 on page 321.

**comparison operators**

Use comparison operators in LOADPLUS predicates to indicate how you want LOADPLUS to compare the left and right sides of predicates.

Comparison rules govern the availability of operators and how processing works. Two sets of comparison rules are available, based on the value of the RULES installation option, as described in the following table:

<table>
<thead>
<tr>
<th>Operators</th>
<th>RULES=STANDARD</th>
<th>RULES=BMC</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;</code></td>
<td>Less than</td>
<td>X</td>
</tr>
<tr>
<td><code>&lt;=</code></td>
<td>Less than or equal</td>
<td>X</td>
</tr>
<tr>
<td><code>=</code></td>
<td>Equal</td>
<td>X</td>
</tr>
<tr>
<td><code>&lt;&gt;</code></td>
<td>Not equal</td>
<td>X</td>
</tr>
<tr>
<td><code>!=</code></td>
<td>Not equal</td>
<td>X</td>
</tr>
<tr>
<td><code>&gt;=</code></td>
<td>Greater than or equal</td>
<td>X</td>
</tr>
<tr>
<td><code>&gt;</code></td>
<td>Greater than</td>
<td>X</td>
</tr>
<tr>
<td><code>IN</code></td>
<td>Equal to any</td>
<td>X</td>
</tr>
<tr>
<td><code>NOT IN</code></td>
<td>Not equal to all</td>
<td>X</td>
</tr>
</tbody>
</table>

**constant**

The right side of a predicate is a constant. Two different sets of rules apply to constants based on the value of the RULES option that is in effect.

**Additional considerations**
Table 58 on page 321 describes the requirements for constants in predicates. In addition, consider the following information:

- Although not prevented or restricted by DB2, you should avoid nonstandard comparisons.
  
  See rules for constants in the IBM DB2 SQL Reference.

- LOADPLUS compares up to 71 bytes of each constant against the field, column, or \( (start:end) \) value.

- LOADPLUS does not validate character data or hexadecimal string data against the current CCSID.

- If you specify a list of constants, note the following information:
  
  — You can improve performance by using ascending sequential order. When you specify the list in this order, you save processing time because LOADPLUS does not sort the list.

  — When you specify \( (start:end) \) with NULLIF or DEFAULTIF, the constants must all be of the same type (for example all character or all hexadecimal) or must all have the same length.
### Table 58: constant requirements for predicates

<table>
<thead>
<tr>
<th>RULES value</th>
<th>Left side of predicate</th>
<th>Constant type</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>STANDARD</td>
<td>fieldName</td>
<td>Character string</td>
<td>Enclose the string in single quotes.</td>
</tr>
</tbody>
</table>
|             |                        | Hexadecimal string | - Enclose the string in single quotes (for example, X'000C').  
|             |                        |               | - The number of bytes that the string represents must match the length of the field data in the input file (SYSREC).  
|             |                        |               | - When the predicate references a Unicode field, the string must be in the internal format of the input. |
|             | Other                  | Other types of constants are not supported. |
| (start:end) | Character string       |               | - Enclose the string in single quotes.  
|             |                        |               | - The length of the string must match the length specified by (start:end). |
|             | Hexadecimal string     |               | - Enclose the string in single quotes (for example, X'000C').  
|             |                        |               | - The number of bytes that the string represents must match the length specified by (start:end).  
|             |                        |               | - When the predicate references a Unicode field, the string must be in the internal format of the input. |
|             | Other                  | Other types of constants are not supported. |
| columnName  | Character string       |               | - Enclose the string in single quotes.  
|             |                        |               | - Precision on a timestamp constant must be less than or equal to the precision defined in the column. |
|             | Hexadecimal string     |               | - Enclose the string in single quotes (for example, X'000C').  
|             |                        |               | - The number of bytes that the string represents must match the length of the field data in the input file (SYSREC).  
<p>|             |                        |               | - When the predicate references a Unicode column, the string must be in the internal format of the input. |
|             | Other                  | Other types of constants are not supported. |</p>
<table>
<thead>
<tr>
<th>RULES value</th>
<th>Left side of predicate</th>
<th>Constant type</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC with <code>fieldName</code> or <code>(start:end)</code></td>
<td><code>fieldName</code></td>
<td>Character string</td>
<td>Enclose the string in single quotes.</td>
</tr>
</tbody>
</table>
| | | Hexadecimal string | - Enclose the string in single quotes (for example, `X'000C'`).  
- When the predicate references a Unicode field, the string must be in the internal format of the output. |
| | | Binary string | Enclose the string in single quotes (for example, `BX'000C'`). |
| | | Other | Other types of constants are not supported. |
| `(start:end)` | Character string | - Enclose the string in single quotes.  
- The length of the string must match the length specified by `(start:end)`.  
- The corresponding table column can be any data type. |
| | Hexadecimal string | - Enclose the string in single quotes (for example, `X'000C'`).  
- The number of bytes that the string represents must match the length specified by `(start:end)`.  
- The corresponding table column can be any data type.  
- When the predicate references a Unicode field, the string must be in the internal format of the output. |
<p>| | Other | Other types of constants are not supported. |</p>
<table>
<thead>
<tr>
<th>RULES value</th>
<th>Left side of predicate</th>
<th>Constant type</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC with <em>columnName</em></td>
<td><em>columnName</em></td>
<td>Integer</td>
<td>The column must be a numeric data type.</td>
</tr>
</tbody>
</table>
| Decimal | | | The column must be a numeric data type.  
**Note:** To determine whether your site has set the decimal point to a comma, check the DB2 subsystem DSNHDECP setting. To enable LOADPLUS to distinguish comma decimal points from comma separators, include a space after any comma that you use as a separator. |
| Character string | | | - Enclose the string in single quotes.  
- The column must be a character, date, time, or timestamp data type.  
- Precision on a timestamp constant must be less than or equal to the precision defined in the column. |
| Hexadecimal string | | | - Enclose the string in single quotes (for example, X’000C’).  
- The column can be any data type.  
- The number of bytes that the string represents must match the length of the column’s internal data. For a VARCHAR or VARBINARY column, the length must be the same length as the maximum length of the column.  
- For a VARCHAR or VARBINARY column, include the two-byte hexadecimal length of the data.  
- Include a preceding null indicator if the column is nullable. For a VARCHAR or VARBINARY column, the length indicator must precede the null indicator.  
- The column must be FIELDPROC encoded.  
- The string must be in DB2 internal format.  
- When the predicate references a Unicode column, the string must be in the internal format of the output. |
<table>
<thead>
<tr>
<th>RULES value with columnName (continued)</th>
<th>Left side of predicate columnName (continued)</th>
<th>Constant type</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC</td>
<td></td>
<td>Graphic string</td>
<td>The column must be a graphic data type. LOADPLUS does not support graphic strings with Unicode data, including the hexadecimal graphic string constants UX’xxxx’ and GX’xxxx’. Note: LOADPLUS does not validate graphic input data to determine whether it contains valid graphic coded character set identifier (GCCSID) values.</td>
</tr>
<tr>
<td>NULL</td>
<td></td>
<td></td>
<td>The column must be nullable. Note: If used with an inequality operator, NULL always collates higher than all other values.</td>
</tr>
<tr>
<td>Binary string</td>
<td>■ Enclose the string in single quotes (for example, BX’000C’). ■ The column must be a binary data type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td>Other constant types are not supported.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1 See Table 51 on page 318 for a description of the difference between field name and column name specifications.

**fieldSpecification**

The field specification options determine the format of the input data.
Field specification (part 1)

```
fieldName
  POSITION (start :end )
  VALUE block
```

Character and graphic data types

```
CHAR (length)
  MIXED
  TRIM
  TRUNCATE

VARCHAR
  MIXED
  TRIM
  TRUNCATE

BLOBF
  PRESERVE WHITESPACE

DBCLOBF
  MIXED
  PRESERVE WHITESPACE

CLOBF
  MIXED

GRAPHIC
  EXTERNAL
  (length)

VARGRAPHIC
```

Numeric and binary data types

```
SMALLINT
  (length, scale)
  ROUND
  REMOVESIGN

INTEGER
  EXTERNAL

BIGINT
  EXTERNAL

DECIMAL
  PACKED
  ZONED
  (length, scale)
  ROUND
  REMOVEDECIMAL
  REMOVESIGN

FLOAT
  EXTERNAL
  (length, scale)
  ROUND

DECFLOAT
  EXTERNAL
  (length)

BINARY
  (length)
  TRIM
  TRUNCATE

VARBINARY
  (length)

BINARY VARYING
```
Field specification (part 2)

Date, time, and timestamp data types

ROWID, LOB, XML, and exit data types
Field specification (part 3)

Note
To see how the field specification syntax fits in the INTO TABLE option, see the diagram in “INTO TABLE” on page 306.

The field specification options include the following specifications:

- Field name, which is usually a column name in the table that you specify
- Starting position of the field value in the input record
- Data type, data format, and data length of the field
- Other processing options

Requirements

The following table describes requirements that apply to field specifications:

Table 59: Field specification requirements

<table>
<thead>
<tr>
<th>Condition</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>You are running a load job that will cause LOADPLUS to invoke DSNUTILB.</td>
<td>Ensure that the field specification is valid for both LOADPLUS and the IBM DB2 LOAD utility. The only exception is that LOADPLUS translates the TRIM option to the appropriate STRIP specification.</td>
</tr>
<tr>
<td>You are specifying POSITION.</td>
<td>Specify POSITION before the data type.</td>
</tr>
<tr>
<td>Condition</td>
<td>Requirement</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>You want to load generated default values into system period start and</td>
<td>Do not include a field specification for these columns.</td>
</tr>
<tr>
<td>end columns or a transaction start column.</td>
<td></td>
</tr>
<tr>
<td>The table contains columns defined as NOT NULL without a DEFAULT defined.</td>
<td>Include either no field specification at all (for any of the columns) or</td>
</tr>
<tr>
<td></td>
<td>include field specifications for all of the NOT NULL columns that do not</td>
</tr>
<tr>
<td></td>
<td>have a DEFAULT defined.</td>
</tr>
<tr>
<td>You specify one of the following options:</td>
<td>Do not include any field specifications.</td>
</tr>
<tr>
<td>■ FORMAT UNLOAD</td>
<td></td>
</tr>
<tr>
<td>■ FORMAT BMC</td>
<td></td>
</tr>
<tr>
<td>■ FORMAT BMCUNLOAD</td>
<td></td>
</tr>
<tr>
<td>You specify FORMAT CSV.</td>
<td>■ Include a field specification for each input field in the order that the</td>
</tr>
<tr>
<td></td>
<td>input field occurs in your input file. The exception to this requirement</td>
</tr>
<tr>
<td></td>
<td>occurs when you do not want to load the last field or fields in the CSV</td>
</tr>
<tr>
<td></td>
<td>file. LOADPLUS ignores any extra input fields at the end of your input file.</td>
</tr>
<tr>
<td></td>
<td>To not load a field other than the last field in the CSV file, include a</td>
</tr>
<tr>
<td></td>
<td>placeholder field specification for that input field.</td>
</tr>
<tr>
<td></td>
<td>■ To have LOADPLUS interpret two consecutive delimiter characters as a null</td>
</tr>
<tr>
<td></td>
<td>field, include the following NULLIF option in your field specification:</td>
</tr>
<tr>
<td></td>
<td>NULLIF fieldName = ''</td>
</tr>
<tr>
<td></td>
<td>Alternatively, you can elect to not include any field specifications. If</td>
</tr>
<tr>
<td></td>
<td>you do not include any field specifications, LOADPLUS loads the data as if</td>
</tr>
<tr>
<td></td>
<td>you had included a field specification that maps to every column that you</td>
</tr>
<tr>
<td></td>
<td>are loading.</td>
</tr>
<tr>
<td>The table contains a row change timestamp column that is defined as</td>
<td>You cannot include this field in your field specification. However, you</td>
</tr>
<tr>
<td>GENERATED ALWAYS.</td>
<td>must include a field specification for at least one of the other input fields.</td>
</tr>
</tbody>
</table>

**Not including field specifications**

When you do not include any field specifications, note the following information:

- LOADPLUS loads the data as if you had included a field specification that maps to every column that is defined on the table.

- LOADPLUS expects your date, time, and timestamp data to be in external format.
LOADPLUS expects the format of all other data to be internal.

**fieldName**

In your field specification, you specify a field name for one of the following purposes:

- To identify the data that you want to load into a column of the table
- As a placeholder in the input record
- To identify a field that you want to use with a predicate of a WHEN, NULLIF, or DEFAULTIF option

If the column corresponding to the named field uses a FIELDPROC, the FIELDPROC must be included in your LINKLIST, JOBLIB, or STEPLIB.

**Requirements**

The field name specification has the following requirements:

- If you use the named field to specify the data that you want to load into a column and you specify SKIPFIELDS NO, the field name must match the corresponding column name.

- If you use the named field only in the predicate of a WHEN, NULLIF, or DEFAULTIF option (and do not want to load data from that field) or only as a placeholder in the input record, the following requirements apply:
  
  — You must specify SKIPFIELDS YES.
  
  — The field name cannot be the name of a column in the table.

  — The data type must be CHAR, VARCHAR, VARGRAPHIC, VARBINARY, or ROWID unless you specify SKIPFIELDS YES ALLTYPES (in which case the field can be any type).

  — Except for variable-length fields, you must supply the length of the field, either with the POSITION(start:end) option or with the length option. The length must be less than or equal to 255.

    If the data type of the field is CHAR, you must explicitly specify a length, even if you use the POSITION option.

    If you use this field specification as a placeholder and you specify FORMAT CSV, you do not need to supply the length of the field. LOADPLUS ignores length specifications with FORMAT CSV.
POSITION(start:end)

This option defines the start (and end) position of the named field. Unless LOADPLUS is invoking DSNUTILB, you can abbreviate POSITION to POS. If you specify POSITION, it must appear before the data type on the field specification.

**Note**

For a DSNUTILB load, if you specify POS instead of POSITION, LOADPLUS terminates.

The variables *start* and *end* designate the locations of the first and last positions of the field. The first position of the record is column one. The start and end positions must be within the bounds of the input record.

**Field length**

The end position determines the length of the field, calculated as *(end - start + 1)*. If you specify *(start,end)* on the POSITION option and you specify the data type, the length that is derived from *(start,end)* must match the explicit or implicit length of the input data type. When there is a conflict between these lengths, LOADPLUS issues message BMC50124E and the job terminates.

If you do not specify POSITION or if you do not specify *end* on the POSITION option when you specify the data type, LOADPLUS uses the field length to determine the length of the input value. BMC recommends that, when you specify the data type, you always specify the length or implied length from the associated column rather than specify *end*. To determine the field length from the data type length, see “Supported data type conversions in LOADPLUS” on page 358.

**CSV data**

When loading CSV data, LOADPLUS currently starts from the first position of each field, ignoring any POSITION *(start,end)* specification. However, BMC recommends that you specify POSITION(*).

In the following descriptions, *n* and *m* are integers.

**start**

This option identifies the starting position of the field. The following table lists the valid values for *start*.
Table 60: Valid start values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>An actual column number; an absolute position in the input record</td>
</tr>
<tr>
<td>*</td>
<td>(default) A relative position one column after the end of the previous field</td>
</tr>
<tr>
<td>* + $n$</td>
<td>A relative position $n + 1$ columns after the end of the previous field</td>
</tr>
</tbody>
</table>

**end**

This option identifies the ending position of the field. The following table lists the valid values for **end**:

Table 61: Valid end values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| $m$   | An actual column number; an absolute position in the input record  
You can use an absolute end position only with an absolute start position. The position that $m$ represents must be greater than or equal to the position that $n$ represents. The length of the field is $m - n + 1$. |
| *     | (default) A relative position one column after the end of the current field  
The length of the field with an end position * and a start position * is 1. |
| * + $m$ | A relative position $m + 1$ columns after the end of the current field  
The position that $m$ represents must be greater than or equal to the position that $n$ represents. You can use an end position * + $m$ only with a start position of * or * + $n$. The length of the field is $m - n + 1$. |

**VALUE**

The **VALUE** option tells LOADPLUS to load a constant value, CURRENT DATE, CURRENT TIME, or CURRENT TIMESTAMP for the column or field.

---

**Note**

You can also use the **VALUE** option with **DEFAULTIF**.

---

**Restrictions**

You cannot specify **VALUE** for a column or field that is defined as any of the following data types:

- **BLOB, CLOB, DBCLOB**
- **XML**
- **ROWID**
  
  For **ROWID**, LOADPLUS uses a unique, generated value as the default.
**constant**

This value supplies a constant value for the specified column or field. Table 62 on page 332 describes the requirements for specifying a constant with the VALUE option.

You can use a hexadecimal string for any data type. Hexadecimal strings have the following requirements:

- The string must be enclosed in single quotes (for example, X’000C’).
- The number of bytes that the string represents must match the length of the column’s internal data. For a VARCHAR or VARBINARY column, the length must be the same length as the maximum length of the column.
- For a VARCHAR or VARBINARY column, the string must include the two-byte hexadecimal length of the data.
- The string must include a preceding null indicator if the column is nullable. For a VARCHAR or VARBINARY column, the length indicator must precede the null indicator.
- The string must be FIELDPROC encoded.
- The string must be in DB2 internal format. When you are loading Unicode data, the string must be in the internal format of the table space.

**Table 62: constant requirements for VALUE**

<table>
<thead>
<tr>
<th>Data type</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numeric</td>
<td>You must use one of the following types of values:</td>
</tr>
<tr>
<td></td>
<td>- Integer</td>
</tr>
<tr>
<td></td>
<td>- Decimal</td>
</tr>
<tr>
<td></td>
<td>- Hexadecimal string (with the requirements listed in this topic)</td>
</tr>
<tr>
<td></td>
<td>- NULL (only if the column is nullable)</td>
</tr>
<tr>
<td>Character</td>
<td>You must use one of the following types of values:</td>
</tr>
<tr>
<td></td>
<td>- Character string, enclosed in single quotes</td>
</tr>
<tr>
<td></td>
<td>- Hexadecimal string (with the requirements listed in this topic)</td>
</tr>
<tr>
<td></td>
<td>- NULL (only if the column is nullable)</td>
</tr>
</tbody>
</table>
### Data type Requirements

<table>
<thead>
<tr>
<th>Data type</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graphic</td>
<td>You must use one of the following types of values:</td>
</tr>
<tr>
<td></td>
<td>■ Graphic string</td>
</tr>
<tr>
<td></td>
<td>■ Hexadecimal string (with the requirements listed in this topic)</td>
</tr>
<tr>
<td></td>
<td>■ NULL (only if the column is nullable)</td>
</tr>
<tr>
<td>Date, time, or timestamp</td>
<td>You must use one of the following types of values:</td>
</tr>
<tr>
<td></td>
<td>■ Character string, enclosed in single quotes</td>
</tr>
<tr>
<td></td>
<td>■ Hexadecimal string (with the requirements listed in this topic)</td>
</tr>
<tr>
<td></td>
<td>■ NULL (only if the column is nullable)</td>
</tr>
<tr>
<td></td>
<td>Precision on a timestamp constant must be less than or equal to the precision defined in the column.</td>
</tr>
<tr>
<td>Binary</td>
<td>You must use one of the following types of values:</td>
</tr>
<tr>
<td></td>
<td>■ Hexadecimal string (with the requirements listed in this topic)</td>
</tr>
<tr>
<td></td>
<td>■ Binary string</td>
</tr>
<tr>
<td></td>
<td>Enclose the string in single quotes (for example, BX'000C').</td>
</tr>
<tr>
<td>Other</td>
<td>Not supported</td>
</tr>
</tbody>
</table>

**CURRENT DATE**

The CURRENT DATE option specifies the current system date. The column to which you are loading this value must be a DATE column.

All CURRENT DATE, TIME, or TIMESTAMP column values for a single row are derived from the same system clock value. For each row, LOADPLUS obtains a new system clock value.

For a two-phase load, LOADPLUS assigns the CURRENT DATE value to the column during the PRELOAD phase, not when actually loading the row during the LOAD phase.

**CURRENT TIME**

The CURRENT TIME option specifies the current system time. The column to which you are loading this value must be a TIME column.

All CURRENT DATE, TIME, or TIMESTAMP column values for a single row are derived from the same system clock value. For each row, LOADPLUS obtains a new system clock value.

For a two-phase load, LOADPLUS assigns the CURRENT TIME value to the column during the PRELOAD phase, not when actually loading the row during the LOAD phase.
CURRENT TIMESTAMP

The CURRENT TIMESTAMP option specifies the current system timestamp. The column to which you are loading this value must be a TIMESTAMP column.

All CURRENT DATE, TIME, or TIMESTAMP column values for a single row are derived from the same system clock value. For each row, LOADPLUS obtains a new system clock value.

For a two-phase load, LOADPLUS assigns the CURRENT TIMESTAMP value to the column during the PRELOAD phase, not when actually loading the row during the LOAD phase.

dataType

The data type option specifies the data type and, in most cases, data format and data length of the input value. For details regarding valid data types and information about using these data types, see “Data type keywords” on page 337.

If you specify options on the data type keywords, you must specify them in the order shown in the syntax diagram.

The following table describes how the data type specification operates:

<table>
<thead>
<tr>
<th>Operation or condition</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defaults</td>
<td>For a field with a corresponding column, if you do not specify a data type, LOADPLUS uses the following defaults:</td>
</tr>
<tr>
<td></td>
<td>■ The default data type is the DB2 data type of the column.</td>
</tr>
<tr>
<td></td>
<td>■ With the exception of date, time, and timestamp fields, the default data format is the internal format of that data type. For date, time, and timestamp fields, the default data format is external.</td>
</tr>
<tr>
<td></td>
<td>For date, time, and timestamp fields, if you specify a data type but not a data format, the default data format is internal.</td>
</tr>
<tr>
<td></td>
<td>■ The default length (or the precision and scale in the case of a DECIMAL column) is the column’s length (or precision and scale).</td>
</tr>
<tr>
<td></td>
<td>For a field with no corresponding column, the default data type is CHARACTER(1).</td>
</tr>
</tbody>
</table>
Data conversion

If the field data type, format, and length are not identical to the column data type, format (always internal), and length, LOADPLUS performs a data conversion. See “Supported data type conversions in LOADPLUS” on page 358 for valid conversions.

If the corresponding column is defined with a FIELDPROC, LOADPLUS applies the FIELDPROC to the input value after any data conversions.

Field length

You can specify a length along with the data type for most data types. In some cases, you must specify a length or POSITION(start:end) to indicate the length of the input field. In other cases, LOADPLUS uses the default length, or you cannot change the length. See “Supported data type conversions in LOADPLUS” on page 358 for the default lengths of the different data types.

CSV data

For CSV input data, LOADPLUS ignores any data type specification other than a date, time, or timestamp external type.

### NULLIF

NULLIF is a field specification option. It determines whether the value that is assigned to a column is null, based on the condition that you specify. If the condition is satisfied, LOADPLUS assigns null to the column. The column must be nullable.

You can specify one of the following values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>predicate</td>
<td>For complete information, see “predicates” on page 316.</td>
</tr>
<tr>
<td>ERROR</td>
<td>ERROR indicates that you want LOADPLUS to assign null to the column if a conversion error occurs. If you specify MSGLEVEL(1) on your EXEC statement or in your installation options, LOADPLUS issues message BMC51501E to indicate the conversion error.</td>
</tr>
</tbody>
</table>

### Restrictions

The following restrictions apply to the NULLIF option:

- You cannot use an identity column in the predicate of a NULLIF statement on the field specification of another column.

- When you specify (start:end), all constants must be of the same type (for example all character or all hexadecimal) or must have the same length.

### Additional considerations

The following additional considerations apply to the NULLIF option:
You can use NULLIF only on a field specification that corresponds to a column.

If you specify multiple NULLIF options, LOADPLUS uses the first matching condition.

The RULES installation option affects the ways in which WHEN, NULLIF, and DEFAULTIF work. For details, see “RULES installation option” on page 819.

**DEFAULTIF**

DEFAULTIF is a field specification option that determines whether the value that is assigned to a column is the default, based on the condition that you specify. If the condition is satisfied, LOADPLUS assigns a default value to the column. The column does not have to be defined NOT NULL WITH DEFAULT.

You can specify one of the following values:

**Table 65: DEFAULTIF option values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>predicate</strong></td>
<td>For complete information, see “predicates” on page 316.</td>
</tr>
<tr>
<td><strong>ERROR</strong></td>
<td>ERROR indicates that you want LOADPLUS to assign the default value to the column if a conversion error occurs. If you specify MSGLEVEL(1) on your EXEC statement or in your installation options, LOADPLUS issues message BMC51501E to indicate the conversion error. <strong>Note:</strong> Use of DEFAULTIF ERROR on a nullable column never results in a null value. The column always defaults to the column’s data type. If you want a null value for a nullable column when a conversion error occurs, use the NULLIF option</td>
</tr>
<tr>
<td><strong>VALUE</strong></td>
<td>VALUE specifies the default value to use. The default DEFAULTIF VALUE is the same as those described by the DB2 SQL CREATE TABLE statement. The syntax for this VALUE option is the same as the syntax for VALUE as it is used for specifying a constant value for a field or column.</td>
</tr>
</tbody>
</table>

**Restrictions**

The following restrictions apply to the DEFAULTIF option:

- LOADPLUS terminates if you specify DEFAULTIF on a LOB or XML column.
- You cannot use an identity column in the predicate of a DEFAULTIF statement on the field specification of another column.
When you specify \((start:end)\), all constants must be of the same type (for example all character or all hexadecimal) or must have the same length.

**Additional considerations**

The following additional considerations apply to DEFAULTIF:

- To obtain a null value with DEFAULTIF on a nullable column, specify DEFAULTIF predicate VALUE(NULL). If you do not specify VALUE with a null indication, the column defaults to the column’s data type. Alternatively, you can use the NULLIF option to obtain the same results.

---

**Note**

You cannot obtain a null value with DEFAULTIF ERROR.

---

- The RULES installation option affects the ways in which WHEN, NULLIF, and DEFAULTIF work. For details, see “RULES installation option” on page 819.

**Data type keywords**

You can use the keywords in the topics that follow to specify data types.

For more information about data types, see the following topics:

**Table 66: Additional information about data types**

<table>
<thead>
<tr>
<th>Information</th>
<th>See reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supported formats for date, time, and timestamp data types</td>
<td>“Supported external DATE TIME and TIMESTAMP formats” on page 356</td>
</tr>
<tr>
<td>Valid data type conversions</td>
<td>“Supported data type conversions in LOADPLUS” on page 358</td>
</tr>
<tr>
<td>Sample data type conversions</td>
<td>“Sample data conversions” on page 376</td>
</tr>
<tr>
<td>Default input data lengths and field lengths</td>
<td>“Supported data type conversions in LOADPLUS” on page 358</td>
</tr>
<tr>
<td>Considerations for numeric fields</td>
<td>“Additional numeric fields considerations” on page 379</td>
</tr>
<tr>
<td>Considerations for date, time, and timestamp fields</td>
<td>“Additional date time and timestamp fields considerations” on page 382</td>
</tr>
<tr>
<td>Considerations for VARCHAR, VARGRAPHIC, VARBINARY, ROWID, and XML fields</td>
<td>“Additional VARCHAR VARGRAPHIC VARBINARY ROWID and XML fields considerations” on page 383</td>
</tr>
</tbody>
</table>
**CHAR**

CHAR (or CHARACTER) defines a fixed-length character field.

**MIXED**

MIXED defines a mixed character data field. You can specify MIXED as either a subtype itself or with the CLOBF subtype.

**BLOBF**

BLOBF indicates that the field contains the name of a binary large object (BLOB) file that you are loading into a LOB or XML column.

When loading CLOB or DBCLOB data from a BLOB file, LOADPLUS invokes DSNUTILB.

**PRESERVE WHITESPACE**

PRESERVE WHITESPACE preserves the white space in the BLOB file when loading into an XML column.

**BINARYXML**

BINARYXML indicates that the referenced file contains binary XML data.

If you also specify the AVGSIZE keyword, you must specify it after BINARYXML.

**AVGSIZE**

This option provides an estimate of the size of the input data. Specify a number, optionally followed by the unit of measure (K, M, or G). The minimum value that you can specify is 1K. The maximum value that you can specify is 2G, 2048M, or 2097152K. The default unit of measure is K.

Alternatively (for LOB input), you can specify a global value with the LOBAVGPCT command or installation option.

**DBCLOBF**

DBCLOBF indicates that the field contains the name of a double-byte character large object (DBCLOB) file that you are loading into a LOB or XML column.
When loading XML, CLOB, or BLOB data from a DBCLOB file, LOADPLUS invokes DSNUTILB.

**PRESERVE WHITESPACE**

PRESERVE WHITESPACE preserves the white space in the DBCLOB file when loading into an XML column.

**AVGSIZE**

This option provides an estimate of the size, in bytes, of the input data. Specify a number, optionally followed by the unit of measure (K, M, or G). The minimum value that you can specify is 1K. The maximum value that you can specify is 2G, 2048M, or 2097152K. The default unit of measure is K.

Alternatively (for LOB input), you can specify a global value with the LOBAVGPCNT command or installation option.

This option is not valid for XML input.

**CLOBF**

CLOBF indicates that the field contains the name of a character large object (CLOB) file that you are loading into a LOB or XML column.

When loading DBCLOB or BLOB data from a CLOB file, LOADPLUS invokes DSNUTILB.

**PRESERVE WHITESPACE**

PRESERVE WHITESPACE tells LOADPLUS to preserve the white space in the CLOB file when loading into an XML column.

**AVGSIZE**

This option provides an estimate of the size of the input data. Specify a number, optionally followed by the unit of measure (K, M, or G). The minimum value that you can specify is 1K. The maximum value that you can specify is 2G, 2048M, or 2097152K. The default unit of measure is K.

Alternatively, for LOB input, you can specify a global value with the LOBAVGPCNT command or installation option. For XML input, you can specify a global value with the XMLAVGSIZE command or installation option.

**(length)**

Length indicates the number of single-byte characters in this field.
**TRIM**

TRIM truncates trailing blanks from the input string value before LOADPLUS attempts to assign the value to a CHAR or VARCHAR column. If the string is still too long, a conversion error occurs. LOADPLUS performs TRIM before TRUNCATE.

**Additional considerations**

The following considerations apply to the TRIM option for a CHAR field:

- For a DSNUTILB load, LOADPLUS translates this option to STRIP TRAILING ' ' before passing the field specification to DSNUTILB.

- TRIM is useful when you are converting from a CHAR field to a VARCHAR column or for converting from a CHAR field to a shorter CHAR column and you want to detect nonblank data beyond position $n$.

- If you are loading Unicode data and do not need to preserve trailing blanks, BMC strongly recommends that you specify TRIM. If translation causes an expansion of the data, specifying this option prevents LOADPLUS from discarding records that you do not want to discard.

**Restriction**

This keyword is not valid with BLOBF, DBCLOBF, or CLOBF subtypes.

**TRUNCATE**

TRUNCATE truncates a string if it is longer than the column length (after trimming). LOADPLUS performs TRUNCATE after TRIM.

When loading Unicode data, LOADPLUS performs truncation after translation.

**Restriction**

This keyword is not valid with BLOBF, DBCLOBF, or CLOBF subtypes.

**VARCHAR**

VARCHAR defines a varying-length character string.

**MIXED**

MIXED defines a mixed character data field. You can specify MIXED as either a subtype itself or with the CLOBF subtype.
**BLOBF**

BLOBF indicates that the field contains the name of a binary large object (BLOB) file that you are loading into a LOB or XML column.

When loading CLOB or DBCLOB data from a BLOB file, LOADPLUS invokes DSNUTILB.

**PRESERVE WHITESPACE**

PRESERVE WHITESPACE preserves the white space in the BLOB file when loading into an XML column.

**BINARYXML**

BINARYXML indicates that the referenced file contains binary XML data.

If you also specify the AVGSIZE keyword, you must specify it after BINARYXML.

**AVGSIZE**

This option provides an estimate of the size of the input data. Specify a number, optionally followed by the unit of measure (K, M, or G). The minimum value that you can specify is 1K. The maximum value that you can specify is 2G, 2048M, or 2097152K. The default unit of measure is K.

Alternatively (for LOB input), you can specify a global value with the LOBAVGPGCT command or installation option. The AVGSIZE field specification overrides the LOBAVGPGCT option.

**DBCLOBF**

DBCLOBF indicates that the field contains the name of a double-byte character large object (DBCLOB) file that you are loading into a LOB or XML column.

When loading XML, CLOB, or BLOB data from a DBCLOB file, LOADPLUS invokes DSNUTILB.

**PRESERVE WHITESPACE**

PRESERVE WHITESPACE preserves the white space in the DBCLOB file when loading into an XML column.

**AVGSIZE**

This option provides an estimate of the size, in bytes, of the input data. Specify a number, optionally followed by the unit of measure (K, M, or G).
The minimum value that you can specify is 1K. The maximum value that you can specify is 2G, 2048M, or 2097152K. The default unit of measure is K.

Alternatively (for LOB input), you can specify a global value with the LOBAVGPT option or installation option. The AVGSIZE field specification overrides the LOBAVGPT option.

This option is not valid for XML input.

**CLOBF**

CLOBF indicates that the field contains the name of a character large object (CLOB) file that you are loading into a LOB or XML column.

When loading DBCLOB or BLOB data from a CLOB file, LOADPLUS invokes DSNUTILB.

**PRESERVE WHITESPACE**

PRESERVE WHITESPACE tells LOADPLUS to preserve the white space in the CLOB file when loading into an XML column.

**AVGSIZE**

This option provides an estimate of the size of the input data. Specify a number, optionally followed by the unit of measure (K, M, or G). The minimum value that you can specify is 1K. The maximum value that you can specify is 2G, 2048M, or 2097152K. The default unit of measure is K.

Alternatively, for LOB input, you can specify a global value with the LOBAVGPT option or installation option. For XML input, you can specify a global value with the XMLAVGSIZE command or installation option. The AVGSIZE field specification overrides the LOBAVGPT and XMLAVGSIZE options.

**TRIM**

TRIM truncates trailing blanks from the input string value before LOADPLUS attempts to assign the value to a CHAR or VARCHAR column. If the string is still too long, a conversion error occurs. LOADPLUS performs TRIM before TRUNCATE.

**Additional considerations**

The following considerations apply to the TRIM option for a VARCHAR field:

- For a DSNUTILB load, LOADPLUS translates this option to STRIP TRAILING '' before passing the field specification to DSNUTILB.
TRIM is useful when converting from a VARCHAR field to a CHAR( n) or VARCHAR( n) column and you want to detect nonblank data beyond position n.

If you are loading Unicode data and do not need to preserve trailing blanks, BMC strongly recommends that you specify TRIM. If translation causes an expansion of the data, specifying this option will prevent LOADPLUS from discarding records that you do not want to discard.

**Restriction**

This keyword is not valid with BLOBF, DBCLOBF, or CLOBF subtypes.

**TRUNCATE**

TRUNCATE truncates a string if it is longer than the column’s maximum length (after trimming). LOADPLUS performs TRUNCATE after TRIM.

When loading Unicode data, LOADPLUS performs truncation after translation.

**Restriction**

This keyword is not valid with BLOBF, DBCLOBF, or CLOBF subtypes.

**GRAPHIC**

GRAPHIC defines a fixed-length graphic field.

Without the EXTERNAL specification, GRAPHIC indicates an internal graphic value without shift-out and shift-in characters.

**GRAPHIC EXTERNAL**

GRAPHIC EXTERNAL indicates an external graphic value with shift-out and shift-in characters.

**(length)**

*Length* indicates the number of double-byte characters. For EXTERNAL, this length does not include the two shift characters.

**VARGRAPHIC**

VARGRAPHIC indicates a varying-length graphic string.
A two-byte length field precedes the string. This length field contains the length of the graphic string. The length is the number of double-byte characters and does not include the two length bytes.

**SMALLINT**

SMALLINT indicates a two-byte internal integer value.

For more information about these options, see “Additional numeric fields considerations” on page 379.

**(length, scale)**

*Length*, if specified, must be 2. You can specify scale with or without specifying a length. Specifying scale multiplies or divides the number by a power of 10.

LOADPLUS does not support the scale specification if you are converting to a DECFLOAT column.

**ROUND**

ROUND tells LOADPLUS to round the value, if necessary, during conversion. If you do not specify ROUND, LOADPLUS truncates any digits that are not needed for precision.

**REMOVESIGN**

When you are converting to a CHAR or VARCHAR column, REMOVESIGN tells LOADPLUS not to include a sign in the values that it loads. When you specify this option, LOADPLUS does not reserve a byte at the beginning of the column for a sign.

**INTEGER**

INTEGER defines an integer field.

You can abbreviate INTEGER to INT. Without the EXTERNAL specification, this data type description indicates a four-byte internal integer value. For more information about these options, see “Additional numeric fields considerations” on page 379.

**INTEGER EXTERNAL**

INTEGER EXTERNAL indicates a string that contains a character representation of a number.
(length, scale)

Length, if specified, must be 4. You can specify scale with or without specifying a length. Specifying scale multiplies or divides the number by a power of 10.

LOADPLUS does not support the scale specification if you are converting to a DECFLOAT column.

ROUND

ROUND tells LOADPLUS to round the value, if necessary, during conversion. If you do not specify ROUND, LOADPLUS truncates any digits that are not needed for precision.

REMOVESIGN

When you are converting to a CHAR or VARCHAR column, REMOVESIGN tells LOADPLUS not to include a sign in the values that it loads. When you specify this option, LOADPLUS does not reserve a byte at the beginning of the column for a sign.

BIGINT

BIGINT indicates an eight-byte internal integer value.

For more information about these options, see “Additional numeric fields considerations” on page 379.

(length, scale)

Length, if specified, must be 8. You can specify scale with or without specifying a length. Specifying scale multiplies or divides the number by a power of 10.

LOADPLUS does not support the scale specification if you are converting to a DECFLOAT column.

ROUND

ROUND tells LOADPLUS to round the value, if necessary, during conversion. If you do not specify ROUND, LOADPLUS truncates any digits that are not needed for precision.
REMOVESIGN

When you are converting to a CHAR or VARCHAR column, REMOVESIGN tells LOADPLUS not to include a sign in the values that it loads. When you specify this option, LOADPLUS does not reserve a byte at the beginning of the column for a sign.

DECIMAL

DECIMAL defines a decimal field.

You can abbreviate DECIMAL to DEC. DECIMAL without an additional keyword functions the same as DECIMAL PACKED. For more information about these options, see “Additional numeric fields considerations” on page 379.

DECIMAL PACKED

DECIMAL PACKED, which is the default, indicates that the input value is packed decimal.

DECIMAL ZONED

DECIMAL ZONED indicates that the input value is zoned decimal.

DECIMAL EXTERNAL

DECIMAL EXTERNAL indicates a string that contains a character representation of a number. If the value includes a decimal point, LOADPLUS ignores any scale specification.

(length, scale)

You can specify one or both of these options. Length (or precision) is the total number of digits. Specifying scale multiplies or divides the number by a power of 10. For more information about scale, see “Scale” on page 379.

ROUND

ROUND tells LOADPLUS to round the value, if necessary, during conversion. If you do not specify ROUND, LOADPLUS truncates any digits that are not needed for precision.
**REMOVEDECIMAL**

When you are converting to a CHAR or VARCHAR column, REMOVEDECIMAL tells LOADPLUS to remove the fractional portion, including the decimal point, of the value that it loads. LOADPLUS performs this function after performing any scale or rounding functions.

**REMOVESIGN**

When you are converting to a CHAR or VARCHAR column, REMOVESIGN tells LOADPLUS not to include a sign in the values that it loads. When you specify this option, LOADPLUS does not reserve a byte at the beginning of the column for a sign.

**FLOAT**

FLOAT defines a floating-point numeric field.

Without the EXTERNAL keyword, this data type description indicates a four- or eight-byte internal floating-point number. For more information, see “Additional numeric fields considerations” on page 379. For examples of data type conversions using FLOAT, see “Sample data conversions” on page 376.

LOADPLUS does not support IEEE Binary Floating Point (BFP) format.

**FLOAT EXTERNAL**

FLOAT EXTERNAL indicates a string that contains a character representation of a number.

*(length, scale)*

*Length*, if specified, must be 1 through 21 for a four-byte internal floating-point number. For an eight-byte internal floating-point number, *length* must be 22 through 53. You can specify scale with or without specifying a length. Specifying *scale* multiplies or divides the number by a power of 10. For more information about *scale*, see “Scale” on page 379.

LOADPLUS does not support the scale specification if you are converting to a DECFLOAT column.

**ROUND**

ROUND tells LOADPLUS to round the value, if necessary, during conversion. If you do not specify ROUND, LOADPLUS truncates any digits that are not needed for precision.
**BINARY**

BINARY defines a fixed-length binary string.

*(length)*

*Length* indicates the number of single-byte characters in this field.

**TRIM**

TRIM truncates trailing binary zeros from the input string value before LOADPLUS attempts to assign the value to a BINARY column. If the string is still too long, a conversion error occurs. LOADPLUS performs TRIM before TRUNCATE.

For a DSNUTILB load, LOADPLUS translates this option to the default DB2 STRIP option before passing the field specification to DSNUTILB.

**TRUNCATE**

TRUNCATE truncates a string if it is longer than the column length (after trimming). LOADPLUS performs TRUNCATE after TRIM.

**VARBINARY**

VARBINARY indicates a varying-length binary string.

A two-byte length field precedes the string. This length field contains the length of the binary string. The length is the number of single-byte characters and does not include the two length bytes.

You can optionally specify this data type as BINARY VARYING.

**TRIM**

TRIM truncates trailing binary zeros from the input string value before LOADPLUS attempts to assign the value to a VARBINARY column. If the string is still too long, a conversion error occurs. LOADPLUS performs TRIM before TRUNCATE.

For a DSNUTILB load, LOADPLUS translates this option to the default DB2 STRIP option before passing the field specification to DSNUTILB.

**TRUNCATE**

TRUNCATE truncates a string if it is longer than the column length (after trimming). LOADPLUS performs TRUNCATE after TRIM.
DATE

DATE defines a date field.

If you specify this option without EXTERNAL, LOADPLUS does not verify input data.

When loading CSV data (FORMAT CSV), you must use the DATE-format EXTERNAL data type, specifying one of the delimited formats listed in “Supported external DATE TIME and TIMESTAMP formats” on page 356.

DATE

DATE indicates a DB2 internal date value.

DATE EXTERNAL(length)

DATE EXTERNAL indicates a DB2 date string value.

DATE-format EXTERNAL(length)

This data type description indicates an external date value. In addition, it indicates the format of the external date value. See “Supported external DATE TIME and TIMESTAMP formats” on page 356 for information about the available date formats.

When you specify DATE-format EXTERNAL, LOADPLUS loads the current date when the input data equates to spaces or zeros unless the column has a default value of null. If the column has a default value of null, LOADPLUS loads nulls instead of the current date when the input data equates to spaces or zeros.

CENTURY(ccyy,ccyy)

CENTURY specifies the 100-year range that determines the century for DATE external formats that contain two-digit year values. The first four-digit year value must be less than the second four-digit year. You must specify both values and these values must span 100 years.

Any two-digit year in the input data that lies between the first yy specification and 99 is prefixed with the first cc value to create a four-digit year. Any two-digit year in the input data that lies between 00 and the second yy specification is prefixed with the second cc value to create a four-digit year.

For example, if you specify CENTURY(1950,2049), LOADPLUS places 19 in front of each two-digit year with a value 50 through 99, and places 20 in front...
of each two-digit year with a value 00 through 49. The date 99/12/31 becomes 1999/12/31 and 00/12/31 becomes 2000/12/31.

For information about specifying CENTURY globally on the LOAD command, see “CENTURY” on page 272. For information about the CENTURY installation option, see “CENTURY=(1950,2049)” on page 717.

**TIMESTAMP**

TIMESTAMP defines a timestamp field.

If you specify this option without EXTERNAL, LOADPLUS does not verify input data.

When loading CSV data (FORMAT CSV), you must specify one of the EXTERNAL TIMESTAMP data types, and the data must be in one of the formats listed in “Supported external DATE TIME and TIMESTAMP formats” on page 356. If you do not specify a format on the field specification, LOADPLUS defaults to format D2E.

**TIMESTAMP**

TIMESTAMP indicates a DB2 internal timestamp value.

**TIMESTAMP EXTERNAL**(length)

TIMESTAMP EXTERNAL indicates a DB2 timestamp string value.

**TIMESTAMP-format EXTERNAL**(length)

This data type description indicates an external timestamp value. In addition, it indicates the format of the external timestamp value. See “Supported external DATE TIME and TIMESTAMP formats” on page 356 for information about the available timestamp formats.

When you specify TIMESTAMP-format EXTERNAL, LOADPLUS loads the current timestamp when the input data equates to spaces or zeros unless the column has a default value of null. If the column has a default value of null, LOADPLUS loads nulls instead of the current timestamp when the input data equates to spaces or zeros.

**CENTURY(ccyy,ccyy)**

CENTURY specifies the 100-year range that determines the century for TIMESTAMP external formats that contain two-digit year values. The first four-digit year value must be less than the second four-digit year. You must specify both values and these values must span 100 years.
Any two-digit year in the input data that lies between the first \(yy\) specification and 99 is prefixed with the first \(cc\) value to create a four-digit year. Any two-digit year in the input data that lies between 00 and the second \(yy\) specification is prefixed with the second \(cc\) value to create a four-digit year.

For example, if you specify CENTURY(1950,2049), LOADPLUS places 19 in front of each two-digit year with a value 50 through 99, and places 20 in front of each two-digit year with a value 00 through 49. The date 99/12/31 becomes 1999/12/31 and 00/12/31 becomes 2000/12/31.

For information about specifying CENTURY globally on the LOAD command, see “CENTURY” on page 272. For information about the CENTURY installation option, see “CENTURY=(1950,2049)” on page 717.

**TIMESTAMP WITH TIME ZONE**

This data type indicates a DB2 internal timestamp with time zone value.

**TIMESTAMP WITH TIME ZONE EXTERNAL** (\(\text{length}\))

This data type indicates a DB2 timestamp with time zone string value.

**TIME**

TIME defines a time field.

If you specify this option without EXTERNAL, LOADPLUS does not verify input data.

When loading CSV data (FORMAT CSV), you must use the TIME-\(\text{format}\) EXTERNAL data type, specifying one of the delimited formats listed in “Supported external DATE TIME and TIMESTAMP formats” on page 356.

**TIME**

TIME indicates a DB2 internal time value.

**TIME EXTERNAL** (\(\text{length}\))

TIME EXTERNAL indicates a DB2 time string value.

**TIME-\(\text{format}\) EXTERNAL** (\(\text{length}\))

This data type description indicates an external time value. In addition, it indicates the format of the external time value. See “Supported external DATE TIME and
TIMESTAMP formats” on page 356 for information about the available time formats.

When you specify TIME-format EXTERNAL(length), LOADPLUS loads the current time when the input data equates to spaces or zeros unless the column has a default value of null. If the column has a default value of null, LOADPLUS loads nulls instead of the current time when the input data equates to spaces or zeros.

ROWID

ROWID defines a field that contains generated row IDs.

This field must be in internal format (a varying length hexadecimal string). The first two bytes of this hexadecimal string contain the length of the row ID string. The length is the number of bytes and does not include the two length bytes.

The following information describes how LOADPLUS processes ROWID fields:

- With the exception described in the restrictions, if the ROWID field is defined as GENERATED ALWAYS, LOADPLUS ignores the VALUE option in the field specification and generates its own unique row identifiers.

- If the ROWID field is defined as GENERATED BY DEFAULT, LOADPLUS performs minimal validation on the ROWID field and loads the ROWID data with no conversions.

Restrictions

ROWID fields in LOADPLUS have the following restrictions:

- You cannot reference a ROWID field in a WHEN, DEFAULTIF, or NULLIF clause.
- You cannot specify DEFAULTIF with VALUE or NULLIF for a ROWID field.
- If the ROWID field is defined as GENERATED ALWAYS and you specify FORMAT BMCLOAD, LOADPLUS discards all rows for the table and issues message BMC51590E.

Note

If you are using LOADPLUS in a CHANGE MANAGER worklist environment, see the CHANGE MANAGER documentation for more information about loading ROWID columns.

BLOB

BLOB indicates a field that contains binary large object (BLOB) data.
A four-byte length field precedes the string. This length does not include the four length bytes.

**Restriction**

LOADPLUS terminates if you specify DEFAULTIF or VALUE on a LOB column.

**AVGSIZE**

This option provides an estimate of the actual size of the input data by specifying a number, optionally followed by the unit of measure (K, M, or G). The minimum value that you can specify is 1K. The maximum values that you can specify are 2G, 2048M, or 2097152K. The default unit of measure is K.

Alternatively, you can specify a global value with the LOBAVGPCT command or installation option. The AVGSIZE field specification overrides the LOBAVGPCT option.

**CLOB**

CLOB indicates a field that contains character large object (CLOB) data.

A four-byte length field precedes the string. This length does not include the four length bytes.

**Restriction**

LOADPLUS terminates if you specify DEFAULTIF or VALUE on a LOB column.

**MIXED**

MIXED indicates that the CLOB input field contains mixed data.

**AVGSIZE**

This option provides an estimate of the actual size of the input data by specifying a number, optionally followed by the unit of measure (K, M, or G). The minimum value that you can specify is 1K. The maximum values that you can specify are 2G, 2048M, or 2097152K. The default unit of measure is K.

Alternatively, you can specify a global value with the LOBAVGPCT command or installation option. The AVGSIZE field specification overrides the LOBAVGPCT option.
DBCLOB

DBCLOB indicates a field that contains double-byte character large object (DBCLOB) data.

A four-byte length field precedes the string. This length does not include the four length bytes.

Restriction

LOADPLUS terminates if you specify DEFAULTIF or VALUE on a LOB column.

AVGSIZE

This option provides an estimate of the actual size of the input data by specifying a number, optionally followed by the unit of measure (K, M, or G). The minimum value that you can specify is 1K. The maximum values that you can specify are 2G, 2048M, or 2097152K. The default unit of measure is K.

Alternatively, you can specify a global value with the LOBAVGPT command or installation option. The AVGSIZE field specification overrides the LOBAVGPT option.

DECFLOAT

DECFLOAT defines a decimal floating-point numeric field.

Without the EXTERNAL keyword, this data type description indicates an 8- or 16-byte internal decimal floating-point number. For more information, see “Additional numeric fields considerations” on page 379.

DECFLOAT EXTERNAL

DECFLOAT EXTERNAL indicates a string that contains a character representation of a number.

(length)

Length, if specified, can be one of the following values:

- If you do not specify EXTERNAL, 16 for an 8-byte number or 34 for a 16-byte number
- If you specify EXTERNAL, 1 through 42
If you do not specify a value or you specify a value greater than 42, LOADPLUS defaults to 42.

**XML**

XML specifies a field that contains XML data.

In a nondelimited input file, a two-byte length field precedes the XML data.

**Restrictions**

LOADPLUS terminates if you specify DEFAULTIF or VALUE on an XML column.

**PRESERVE WHITESPACE**

PRESERVE WHITESPACE tells LOADPLUS to preserve the white space in the XML column.

**BINARYXML**

BINARYXML indicates that the input data is binary XML data.

**AVGSIZE**

This option provides an estimate of the actual size of the input data by specifying a number, optionally followed by the unit of measure (K, M, or G). The minimum value that you can specify is 1K. The maximum values that you can specify are 2G, 2048M, or 2097152K. The default unit of measure is K.

Alternatively, you can specify a global value with the XMLAVGSIZE command or installation option. The AVGSIZE field specification overrides the XMLAVGSIZE option.

**EXIT**

EXIT allows you to request your own data conversion by using a user-written exit routine.

You must specify POSITION(start:end) or the length option to define the input field length.

**programName**

This value is the name of your conversion exit routine. You must include the library that contains this program in your LINKLIST, JOBLIB, or STEPLIB.
The following table shows the return codes that LOADPLUS expects in register 15:

### Table 67: User exit return codes

<table>
<thead>
<tr>
<th>Return code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Conversion complete</td>
</tr>
</tbody>
</table>
| 4           | Conversion error  
LOADPLUS issues message BMC50289E and rejects this record but continues processing other records. If you specified the DISCARDS option, records that LOADPLUS discards because of a conversion error count toward the discards limit. |
| 8           | Parameter error  
LOADPLUS issues message BMC50289E and does not process other records. The utility return code is set to 8. |

If the exit returns an invalid return code, LOADPLUS issues message BMC50289E and terminates. A sample exit is located in member AMUXDEPT of the HLQ.LLQSAMP data set (where HLQ is the high-level qualifier specified during installation and LLQ is the low-level qualifier or prefix set during installation).

**PARM (constant1, constant2,...)**

You can specify an optional parameter list that contains integer, decimal, string, or hexadecimal constants. LOADPLUS does not support floating-point and graphic constants.

The parameter list that you pass to a conversion exit routine is identical to the parameter list that is passed to a DB2 FIELDPROC exit routine; that is, the FPBFCODE in the FPIB is set to FBBFENC (= 0), the CVD is the value from the input record, the FVD is the converted column value, and the FPPVL is the PARM.

Coding requirements for the LOADPLUS conversion exit are the same as those for a DB2 FIELDPROC. For more information, see the IBM DB2 documentation.

**Supported external DATE TIME and TIMESTAMP formats**

The tables in this topic list input date, time, and timestamp formats, other than internal DB2 formats, that LOADPLUS supports.

The format value listed in these tables is the value that you use in place of the `format` in the `DATE-format EXTERNAL`, `TIME-format EXTERNAL`, and `TIMESTAMP-format EXTERNAL` data type specification.
# External date formats

Table 68: Date formats

<table>
<thead>
<tr>
<th>Format value Char/Int/Dec</th>
<th>Non delimited format</th>
<th>Format value Char</th>
<th>Delimited format&lt;sup&gt;a&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 I1 P1</td>
<td>MMDDYY</td>
<td>D1</td>
<td>MM/DD/YY</td>
</tr>
<tr>
<td>2 I2 P2</td>
<td>YYMMDD</td>
<td>D2</td>
<td>YY/MM/DD</td>
</tr>
<tr>
<td>3 I3 P3</td>
<td>YYDDD</td>
<td>D3</td>
<td>YY/DDD</td>
</tr>
<tr>
<td>4 I4 P4</td>
<td>YYWWD</td>
<td>D4</td>
<td>YY/WW/D</td>
</tr>
<tr>
<td>5 I5 P5</td>
<td>DDMMYY</td>
<td>D5</td>
<td>DD/MM/YY</td>
</tr>
<tr>
<td>6 I6 P6</td>
<td>DDDDD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1E I1E P1E</td>
<td>MMDDYYYY</td>
<td>D1E</td>
<td>MM/DD/YYYY</td>
</tr>
<tr>
<td>2E I2E P2E</td>
<td>YYYYMMDD</td>
<td>D2E</td>
<td>YYYY/MM/DD</td>
</tr>
<tr>
<td>3E I3E P3E</td>
<td>YYYYDDD</td>
<td>D3E</td>
<td>YYYY/DDD</td>
</tr>
<tr>
<td>4E I4E P4E</td>
<td>YYYYWWD</td>
<td>D4E</td>
<td>YYYY/WW/D</td>
</tr>
<tr>
<td>5E I5E P5E</td>
<td>DDMMYYYY</td>
<td>D5E</td>
<td>DD/MM/YYYY</td>
</tr>
</tbody>
</table>

1 '/' can be any nonnumeric character. However, for FORMAT CSV, do not use delimiting characters that you specify with TERMINATEDBY or ENCLOSEDBY.

# External time formats

Table 69: Time formats

<table>
<thead>
<tr>
<th>Format value Char/Int/Dec</th>
<th>Non delimited format</th>
<th>Format value Char</th>
<th>Delimited format&lt;sup&gt;a&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 I1 P1</td>
<td>HHMMSS</td>
<td>D1</td>
<td>HH.MM.SS</td>
</tr>
<tr>
<td>2 I2 P2</td>
<td>HHMM</td>
<td>D2</td>
<td>HH.MM</td>
</tr>
<tr>
<td>Not applicable</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

1 '.' can be any nonnumeric character. However, for FORMAT CSV, do not use delimiting characters that you specify with TERMINATEDBY or ENCLOSEDBY.

2 xM is either AM or PM.
### External timestamp formats

Any of the following timestamp formats except format 3 can include the time zone offset in the format \( \pm HH:MM \).

#### Table 70: Timestamp formats

<table>
<thead>
<tr>
<th>Format value</th>
<th>Nondelimited format</th>
<th>Format value</th>
<th>Delimited format (^a)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>YYMMDDHHMMSS</td>
<td>D1</td>
<td>YY-MM-DD-HH.MM.SS</td>
</tr>
<tr>
<td>2</td>
<td>YYMMDDHHMMSSNNNNNN(^b)</td>
<td>D2</td>
<td>YY-MM-DD-HH.MM.SS.NNNNNN(^b)</td>
</tr>
<tr>
<td>3</td>
<td>Store clock 8-byte binary</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>1E</td>
<td>YYYYMMDDHHMMSS</td>
<td>D1E</td>
<td>YYYY-MM-DD-HH.MM.SS</td>
</tr>
<tr>
<td>2E</td>
<td>YYYYMMDDHHMMSSNNNNNN(^b)</td>
<td>D2E</td>
<td>YYYY-MM-DD-HH.MM.SS.NNNNNN(^b)</td>
</tr>
</tbody>
</table>

\(^a\) can be any nonnumeric character except delimiting characters that you specify with TERMINATEDBY or ENCLOSEDBY

\(^b\) NNNNNNN can be any number of microseconds from one through twelve. For FORMAT CSV, NNNNNNN must be less than or equal to the precision defined in the column that you are loading.

---

### Supported data type conversions in LOADPLUS

This topic describes the allowable data conversions and default output lengths.

A blank cell in a table indicates that the data type conversion is not allowed.

#### Numeric output columns

The following table lists the conversions to numeric output that LOADPLUS supports. This table indicates the supported conversions with the following identifiers:

- X = allowable conversion
- D = allowable conversion and the default input data type
Note
When you specify FORMAT CSV, LOADPLUS supports only conversions to the following data types:
- DATE EXTERNAL
- TIME EXTERNAL
- TIMESTAMP EXTERNAL

Table 71: Allowable data type conversions for numeric output columns

<table>
<thead>
<tr>
<th>Output column</th>
<th>SMALLINT</th>
<th>INTEGER</th>
<th>BIGINT</th>
<th>DECIMAL</th>
<th>FLOAT</th>
<th>DECFLOAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td>D</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>INTEGER</td>
<td>X</td>
<td>D</td>
<td>X</td>
<td>X</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>INTEGER EXTERNAL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>BIGINT</td>
<td>X</td>
<td>X</td>
<td>D</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>D</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>D</td>
<td>X</td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHAR BLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR CLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR DBCLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VARCHAR BLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR CLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output column</td>
<td>SMALLINT</td>
<td>INTEGER</td>
<td>BIGINT</td>
<td>DECIMAL</td>
<td>FLOAT</td>
<td>DECFLOAT</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------</td>
<td>---------</td>
<td>--------</td>
<td>---------</td>
<td>-------</td>
<td>----------</td>
</tr>
<tr>
<td><strong>Input field</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE-<code>format</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-<code>format</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP-<code>format</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XML</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXIT</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
1. This conversion is inexact. The value in the floating-point column might not be identical to the value in the input field.

2. The format number describes the representation of the input date or time value. If you do not specify the format number, LOADPLUS defaults to the format specified in your DSNHDECN module in most cases (ISO, USA, EUR, JIS, or LOCAL). However, for CSV input data, LOADPLUS defaults to ISO format.

The following table describes the default length of values in your input file based on data type. This table uses the following identifiers:

- Numeric values = required input length
- X = requires explicit length
- C = column defines default length
  For DECIMAL, the column also defines precision and scale.
- V = input value defines the length; two-byte or four-byte length precedes the field value

Table 72: Default lengths of input values

<table>
<thead>
<tr>
<th>Output column</th>
<th>SMALLINT</th>
<th>INTEGER</th>
<th>BIGINT</th>
<th>DECIMAL</th>
<th>FLOAT</th>
<th>DECFLOAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>INTEGER</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>INTEGER EXTERNAL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>BIGINT</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>C</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>C</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>FLOAT</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>C</td>
<td>X</td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Output column</td>
<td>SMALLINT</td>
<td>INTEGER</td>
<td>BIGINT</td>
<td>DECIMAL</td>
<td>FLOAT</td>
<td>DECFLOAT</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------</td>
<td>---------</td>
<td>--------</td>
<td>---------</td>
<td>-------</td>
<td>----------</td>
</tr>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>V</td>
<td>V</td>
<td>V</td>
<td>V</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE-format EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-format EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP-format EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XML</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Binary, character, and graphic output columns

The following table lists the conversions that LOADPLUS supports to binary, character, and graphic output columns. This table indicates the supported conversions with the following identifiers:

- **X** = allowable conversion
- **D** = allowable conversion and the default input data type

**Note**

When you specify FORMAT CSV, LOADPLUS supports only conversions to the following data types:

- DATE EXTERNAL
- TIME EXTERNAL
- TIMESTAMP EXTERNAL

#### Table 73: Allowable data type conversions for binary, character, and graphic output columns

<table>
<thead>
<tr>
<th>Output column</th>
<th>BINARY</th>
<th>VARBINARY</th>
<th>CHARACTER</th>
<th>VARCHAR</th>
<th>GRAPHIC</th>
<th>VARGRAPHIC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input field</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td>D</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td>X</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output column</td>
<td>BINARY</td>
<td>VARBINARY</td>
<td>CHARACTER</td>
<td>VARCHAR</td>
<td>GRAPHIC</td>
<td>VARGRAPHIC</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------</td>
<td>-----------</td>
<td>-----------</td>
<td>---------</td>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td>D</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR BLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR CLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR DBCLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td>X</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR BLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR CLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR DBCLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td>D</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC EXTERNAL</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td>X</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE-format a EXTERNAL</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-format a EXTERNAL</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP-format a EXTERNAL</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

LOADPLUS for DB2 Reference Manual
The format number describes the representation of the input date or time value. If you do not specify the format number, LOADPLUS defaults to the format specified in your DSNHDECP module in most cases (ISO, USA, EUR, JIS, or LOCAL). However, for CSV input data, LOADPLUS defaults to ISO format.

The following table describes the default length of values in your input file based on data type. This table uses the following identifiers:

- Numeric values = required input length
- X = requires explicit length
- C = column defines default length
  For DECIMAL, the column also defines precision and scale.
  For GRAPHIC EXTERNAL, this value includes the shift-in and shift-out characters.
- V = input value defines the length; two-byte or four-byte length precedes the field value
- T = tables in “Supported external DATE TIME and TIMESTAMP formats” on page 356 provide length information.

Table 74: Default lengths of input values

<table>
<thead>
<tr>
<th>Output column</th>
<th>BINARY</th>
<th>VARBINARY</th>
<th>CHARACTER</th>
<th>VARCHAR</th>
<th>GRAPHIC</th>
<th>VARGRAPHIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XML</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXIT</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

1 The format number describes the representation of the input date or time value. If you do not specify the format number, LOADPLUS defaults to the format specified in your DSNHDECP module in most cases (ISO, USA, EUR, JIS, or LOCAL). However, for CSV input data, LOADPLUS defaults to ISO format.
<table>
<thead>
<tr>
<th>Output column</th>
<th>BINARY</th>
<th>VARBINARY</th>
<th>CHARACTER</th>
<th>VARCHAR</th>
<th>GRAPHIC</th>
<th>VARGRAPHIC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input field</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER EXTERNAL</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td>C</td>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td>V</td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td></td>
<td>C</td>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td>V</td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>GRAPHIC EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE-format EXTERNAL</td>
<td></td>
<td></td>
<td>T</td>
<td>T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-format EXTERNAL</td>
<td></td>
<td></td>
<td>T</td>
<td>T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP-format EXTERNAL</td>
<td></td>
<td></td>
<td>T</td>
<td>T</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Date, time, and timestamp output columns

The following table lists the conversions that LOADPLUS supports to date, time, and timestamp output columns. This table indicates the supported conversions with the following identifiers:

- X = allowable conversion
- D = allowable conversion and the default input data type

**Note**
When you specify FORMAT CSV, LOADPLUS supports only conversions to the following data types:
- DATE EXTERNAL
- TIME EXTERNAL
- TIMESTAMP EXTERNAL

<table>
<thead>
<tr>
<th>Output column</th>
<th>BINARY</th>
<th>VARBINARY</th>
<th>CHARACTER</th>
<th>VARCHAR</th>
<th>GRAPHIC</th>
<th>VARGRAPHIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
<td>33</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XML</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXIT</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 75: Allowable data type conversions for date, time, and timestamp output columns

<table>
<thead>
<tr>
<th>Output column</th>
<th>DATE</th>
<th>TIME</th>
<th>TIMESTAMP</th>
<th>TIMESTAMP WITH TIME ZONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER EXTERNAL</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output column</td>
<td>DATE</td>
<td>TIME</td>
<td>TIMESTAMP</td>
<td>TIMESTAMP WITH TIME ZONE</td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
<td>------</td>
<td>-----------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHAR BLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR CLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR DBCLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VARCHAR BLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR CLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR DBCLOBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE-format a</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-format a</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE WITH TIME ZONE</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
The format number describes the representation of the input date or time value. If you do not specify the format number, LOADPLUS defaults to the format specified in your DSNHDECP module in most cases (ISO, USA, EUR, JIS, or LOCAL). However, for CSV input data, LOADPLUS defaults to ISO format.

The following table describes the default length of values in your input file based on data type. This table uses the following identifiers:

- **Numeric values** = required input length
- **X** = requires explicit length
- **V** = input value defines the length; two-byte or four-byte length precedes the field value
- **L** = local DATE and TIME length if your DSNHDECP format is LOCAL. If your DSNHDECP format is ISO, USA, EUR, or JIS, the default length for DATE is 10 and the default length for TIME is 8.
- **T** = tables in “Supported external DATE TIME and TIMESTAMP formats” on page 356 provide length information.

### Table 76: Default lengths of input values

<table>
<thead>
<tr>
<th>Output column</th>
<th>DATE</th>
<th>TIME</th>
<th>TIMESTAMP</th>
<th>TIMESTAMP WITH TIME ZONE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input field</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output column</td>
<td>DATE</td>
<td>TIME</td>
<td>TIMESTAMP</td>
<td>TIMESTAMP WITH TIME ZONE</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------</td>
<td>------</td>
<td>-----------</td>
<td>--------------------------</td>
</tr>
<tr>
<td><strong>Input field</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>4</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER EXTERNAL</td>
<td>8</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td>8</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td>8</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td>8</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td></td>
<td>8</td>
<td>26</td>
<td>33</td>
</tr>
<tr>
<td>DATE-format EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-format EXTERNAL</td>
<td>T</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td>26</td>
<td>26</td>
<td>26</td>
<td>26</td>
</tr>
<tr>
<td>TIMESTAMP-format EXTERNAL</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td>33</td>
<td>33</td>
<td>33</td>
<td>33</td>
</tr>
</tbody>
</table>
The following table lists the conversions that LOADPLUS supports to LOB and XML output columns. This table indicates the supported conversions with the following identifiers:

- X = allowable conversion
- D = allowable conversion and the default input data type

**Note**
When you specify FORMAT CSV, LOADPLUS supports only conversions to the following data types:
- DATE EXTERNAL
- TIME EXTERNAL
- TIMESTAMP EXTERNAL

### Table 77: Allowable data type conversions for LOB and XML output columns

<table>
<thead>
<tr>
<th>Output column</th>
<th>BLOB</th>
<th>CLOB</th>
<th>DBCLOB</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output column</td>
<td>BLOB</td>
<td>CLOB</td>
<td>DBCLOB</td>
<td>XML</td>
</tr>
<tr>
<td>---------------------</td>
<td>------</td>
<td>------</td>
<td>--------</td>
<td>-----</td>
</tr>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR BLOBF</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>CHAR CLOBF</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>CHAR DBCLOBF</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR BLOBF</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>VARCHAR CLOBF</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>VARCHAR DBCLOBF</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE-format</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-format</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP-format</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
LOADPLUS invokes DSNUTILB in this case.

The format number describes the representation of the input date or time value. If you do not specify the format number, LOADPLUS defaults to the format specified in your DSNHDECP module in most cases (ISO, USA, EUR, JIS, or LOCAL). However, for CSV input data, LOADPLUS defaults to ISO format.

The following table describes the default length of values in your input file based on data type. The V in this table indicates that the input value defines the length and a two-byte or four-byte length precedes the field value.

**Table 78: Default lengths of input values**

<table>
<thead>
<tr>
<th>Output column</th>
<th>BLOB</th>
<th>CLOB</th>
<th>DBCLOB</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>XML</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>EXIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output column</td>
<td>BLOB</td>
<td>CLOB</td>
<td>DBCLOB</td>
<td>XML</td>
</tr>
<tr>
<td>---------------</td>
<td>------</td>
<td>------</td>
<td>--------</td>
<td>-----</td>
</tr>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE-format EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-format EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP-format EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td>V</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>XML</td>
<td></td>
<td></td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>EXIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Additional input field length information

The table in this topic describes additional information about input field lengths for some data types.

**Table 79: Additional field length information**

<table>
<thead>
<tr>
<th>Field data type</th>
<th>Field length (in bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER EXTERNAL(n)</td>
<td>n</td>
</tr>
<tr>
<td>DECIMAL(n)</td>
<td>Integer(n/2) + 1</td>
</tr>
<tr>
<td>DECIMAL EXTERNAL(n)</td>
<td>n</td>
</tr>
<tr>
<td>DECIMAL ZONED(n)</td>
<td>n</td>
</tr>
<tr>
<td>FLOAT(n)</td>
<td>4 if 1 (\leq n \leq 21)</td>
</tr>
<tr>
<td></td>
<td>8 if 22 (\leq n \leq 53)</td>
</tr>
<tr>
<td>FLOAT EXTERNAL(n)</td>
<td>n</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>Length is determined by the two-byte length + 2</td>
</tr>
<tr>
<td>GRAPHIC(n)</td>
<td>2n</td>
</tr>
<tr>
<td>GRAPHIC EXTERNAL(n)</td>
<td>2n + 2 for the shift characters</td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td>Length is determined by the two-byte length * 2 + 2</td>
</tr>
<tr>
<td>DATE EXTERNAL(n)</td>
<td>n</td>
</tr>
<tr>
<td>DATE-format EXTERNAL(n)</td>
<td>n if CHARACTER</td>
</tr>
<tr>
<td></td>
<td>Always 4 if INTEGER</td>
</tr>
<tr>
<td></td>
<td>Integer(n/2) + 1 if DECIMAL</td>
</tr>
<tr>
<td>TIME EXTERNAL(n)</td>
<td>n</td>
</tr>
<tr>
<td>TIME-format EXTERNAL(n)</td>
<td>n if CHARACTER</td>
</tr>
<tr>
<td></td>
<td>Always 4 if INTEGER</td>
</tr>
<tr>
<td></td>
<td>Integer(n/2) + 1 if DECIMAL</td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL(n)</td>
<td>n</td>
</tr>
<tr>
<td>TIMESTAMP-format EXTERNAL(n)</td>
<td>n if CHARACTER</td>
</tr>
<tr>
<td></td>
<td>Always 4 if INTEGER</td>
</tr>
<tr>
<td></td>
<td>Integer(n/2) + 1 if DECIMAL</td>
</tr>
<tr>
<td>ROWID</td>
<td>Length is determined by the two-byte length + 2</td>
</tr>
<tr>
<td>BLOB</td>
<td>Length is determined by the four-byte length + 4</td>
</tr>
<tr>
<td>CLOB</td>
<td>Length is determined by the four-byte length + 4</td>
</tr>
<tr>
<td>DBCLOB</td>
<td>Length is determined by the four-byte length + 4</td>
</tr>
</tbody>
</table>
Sample data conversions

The tables in this topic provide sample character, numeric, date, time, and timestamp conversions.

These tables use the following standard information:

- In the **Data input** and **Data column** values
  - Character strings are depicted as '\texttt{string}'.
  - The letter \texttt{b} represents a blank in the character strings.
  - Hexadecimal strings are depicted as \texttt{X'hexString'}.
  - Noncharacter input and column values are depicted as hexadecimal strings.

- In the **Data column** values
  - All column values are depicted in non-normalized DB2 internal format.
  - All columns are defined NOT NULL and no column values contain the null indicator.

Table 80: Sample character data conversions

<table>
<thead>
<tr>
<th>Data type input</th>
<th>Data type column</th>
<th>Data input</th>
<th>Data column</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>CHAR(8)</td>
<td>'ABCDEFbb'</td>
<td>'ABCDEFbb'</td>
<td>Input length defaults to column length</td>
</tr>
<tr>
<td>CHAR(6)</td>
<td>CHAR(8)</td>
<td>'ABCDEF'</td>
<td>'ABCDEFbb'</td>
<td>Result padded with blanks</td>
</tr>
<tr>
<td>CHAR(8)</td>
<td>VARCHAR(8)</td>
<td>'ABCDEFbb'</td>
<td>X'0008','ABCDEFbb' or 'ABCDEFbb'</td>
<td>Basic row format or Reordered row format</td>
</tr>
<tr>
<td>CHAR(8) TRIM</td>
<td>VARCHAR(6)</td>
<td>'ABCDEFbb'</td>
<td>X'0006','ABCDEF' or 'ABCDEF'</td>
<td>Basic row format or Reordered row format Trailing blanks truncated</td>
</tr>
<tr>
<td>Data type input</td>
<td>Data type column</td>
<td>Data input</td>
<td>Data column</td>
<td>Comments</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>CHAR(8)</td>
<td>VARCHAR(6)</td>
<td>'ABCDEFbb'</td>
<td>Conversion error</td>
<td>Target too small</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>CHAR(8)</td>
<td>X'0004';'ABCD'</td>
<td>'ABCDbbbb'</td>
<td>Result padded with blanks</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>VARCHAR(6)</td>
<td>X'0008';'ABCDEFGH'</td>
<td>Conversion error</td>
<td>Target too small</td>
</tr>
</tbody>
</table>

**Table 81: Sample numeric data conversions**

<table>
<thead>
<tr>
<th>Data type input</th>
<th>Data type column</th>
<th>Data input</th>
<th>Data column</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>INTEGER</td>
<td>'bbbbbbbb10'</td>
<td>X'0000000A'</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>INTEGER</td>
<td>'bbbb10bbbb'</td>
<td>X'00000064'</td>
<td>Result = 100</td>
</tr>
<tr>
<td>INTEGER</td>
<td>INTEGER(2)</td>
<td>X'00000064'</td>
<td>X'00000001'</td>
<td>Input value = 100</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Result = 1</td>
</tr>
<tr>
<td>INTEGER</td>
<td>CHAR(10)</td>
<td>X'00000064'</td>
<td>'bbbbbbbb100'</td>
<td>Input value = 100</td>
</tr>
<tr>
<td>INTEGER(2)</td>
<td>DECIMAL(5,2)</td>
<td>X'0000007D'</td>
<td>X'00125C'</td>
<td>Input value = 125</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Result = 1.25</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>DECIMAL(5,2)</td>
<td>'12345'</td>
<td>X'12345C'</td>
<td>Result = 123.45</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>DECIMAL(7,2)</td>
<td>'bb12345'</td>
<td>X'1234500C'</td>
<td>Result = 12345.00</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>DECIMAL(8,2)</td>
<td>'12345678.b'</td>
<td>Conversion error</td>
<td>Target too small</td>
</tr>
<tr>
<td>DECIMAL(10,4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL(5,1)</td>
<td>INTEGER</td>
<td>'bb15b'</td>
<td>X'00000001'</td>
<td>Fraction truncated</td>
</tr>
<tr>
<td>DECIMAL(5,2)</td>
<td>DECIMAL(5,2)</td>
<td>'-1234'</td>
<td>X'01234D'</td>
<td>Result = -12.34</td>
</tr>
<tr>
<td>DECIMAL(5,1)</td>
<td>DECIMAL(5)</td>
<td>'bbb15'</td>
<td>X'000002C'</td>
<td>Result = 2 due to rounding</td>
</tr>
<tr>
<td>DECIMAL ZONED(4,0)</td>
<td>DECIMAL(4,0)</td>
<td>X'F1F2F3D3'</td>
<td>X'01233D'</td>
<td>Result = -1233.</td>
</tr>
<tr>
<td>DECIMAL ZONED(4,0)</td>
<td>DECIMAL(4,0)</td>
<td>X'F1F2F3C4'</td>
<td>X'01234C'</td>
<td>Result = 1234.</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>DECIMAL(5,2)</td>
<td>X'12345C'</td>
<td>X'12345C'</td>
<td>Input and result = 123.45</td>
</tr>
<tr>
<td>Data type input</td>
<td>Data type column</td>
<td>Data input</td>
<td>Data column</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------------</td>
<td>----------------</td>
<td>-----------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>DECIMAL(5,2)</td>
<td>DECIMAL(4,2)</td>
<td>X'12345C'</td>
<td>Conversion error</td>
<td>Target too small</td>
</tr>
<tr>
<td>FLOT(53)</td>
<td>DECIMAL(5,1)</td>
<td>X'40DC28F5C28F5C28'</td>
<td>X'00859C'</td>
<td>Input = . 8599999999999999 E+0</td>
</tr>
<tr>
<td>FLOT(53)</td>
<td>DECIMAL(7,4)</td>
<td>X'4239C65B1343EC2E'</td>
<td>X'0577748C'</td>
<td>Input = . 57774781421740208 E+2 ROUND is specified</td>
</tr>
</tbody>
</table>

1 For more examples of INTEGER to CHAR conversions, see "Converting to character data" on page 372.

### Table 82: Sample date, time, and timestamp data conversions

<table>
<thead>
<tr>
<th>Data type input</th>
<th>Data type column</th>
<th>Data input</th>
<th>Data column</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>No data type</td>
<td>DATE</td>
<td>'01/31/1990'</td>
<td>X'19900131'</td>
<td>Input data type defaults to DATE EXTERNAL</td>
</tr>
<tr>
<td>DATE</td>
<td>DATE</td>
<td>X'19900131'</td>
<td>X'19900131'</td>
<td>Day out of range</td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td>DATE</td>
<td>'1990-2-29b'</td>
<td>Conversion error</td>
<td>Day out of range</td>
</tr>
<tr>
<td>DATE-2 EXTERNAL CENTURY (1950-2049)</td>
<td>DATE</td>
<td>'900131'</td>
<td>X'19900131'</td>
<td>Year '90' becomes 1990 with CENTURY specification</td>
</tr>
<tr>
<td>DATE-D1 EXTERNAL CENTURY (1950-2049)</td>
<td>DATE</td>
<td>'01/31/90'</td>
<td>X'19900131'</td>
<td>Year '00' becomes 2000 with CENTURY specification</td>
</tr>
<tr>
<td>DATE-P1E EXTERNAL</td>
<td>DATE</td>
<td>X'002291990C'</td>
<td>Conversion error</td>
<td>Day out of range Input value = 2/29/1990</td>
</tr>
<tr>
<td>TIME</td>
<td>TIME</td>
<td>X'123000'</td>
<td>'12:30 PMbb'</td>
<td></td>
</tr>
<tr>
<td>TIME-I1 EXTERNAL</td>
<td>TIME</td>
<td>X'0001E078'</td>
<td>X'123000'</td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL(19)</td>
<td>TIMESTAMP</td>
<td>'1990-1-31-12.30.00b'</td>
<td>X'1990013112300000000'</td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL(26)</td>
<td>TIMESTAMP</td>
<td>'1990-01-31-12.30.00.000000'</td>
<td>X'19900131123000000000'</td>
<td></td>
</tr>
</tbody>
</table>
### Additional numeric fields considerations

This topic describes additional information for numeric input data.

**Numeric EXTERNAL**

A numeric EXTERNAL field consists of a valid integer, decimal, or floating-point EBCDIC constant as defined by DB2 SQL. LOADPLUS allows leading and trailing blanks. With the exception of DECFLOAT, you must specify a length or POSITION(start:end). For DECFLOAT, if you do not specify a length, LOADPLUS defaults to 42. A length specification indicates the number of characters in the field, not the precision. The scale defaults to 0 if the input data does not contain an explicit decimal point.

**Scale**

You can specify scale on integer, FLOAT, and DECIMAL fields with or without specifying a length. Specifying scale multiplies or divides the number by a power of 10. For example, a scale of 3 divides the number by 1000 while a scale of -2 multiplies the number by 100.

**Additional information**

The following additional information applies to the scale specification:

- If you specify scale on a DECIMAL EXTERNAL field, LOADPLUS ignores the specification for input values that contain a decimal point.

- LOADPLUS performs the scale function after any rounding but before any REMOVEDECIMAL and REMOVESIGN functions.

- If you do not specify scale, 0 is the default unless the column is DECIMAL and the field is DECIMAL(PACKED) (explicit or defaulted), or DECIMAL ZONED. In these cases, the scale defaults to the scale of the column.

---

<table>
<thead>
<tr>
<th>Data type input</th>
<th>Data type column</th>
<th>Data input</th>
<th>Data column</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIMESTAMP-3 EXTERNAL</td>
<td>TIMESTAMP</td>
<td>X'197DC8F1C2000000'</td>
<td>X'1990013112300000</td>
<td>Input is 8-byte store clock value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'0000'</td>
<td>0000'</td>
<td>Result = 1990-01-31-12.30.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TIMESTAMP</th>
<th>CHAR(19)</th>
<th>X'19900131123000000000'</th>
<th>'1990-01-31-12.30.00'</th>
<th></th>
</tr>
</thead>
</table>
With the exception of a decimal field, LOADPLUS does not support the scale specification on an input field if you are converting to a DECFLOAT column.

**Rounding and truncation**

With the exception of DECFLOAT, if you specify ROUND, LOADPLUS rounds the value during conversion to fit into the target column. If you do not specify ROUND, LOADPLUS truncates any additional digits that are not needed for precision. LOADPLUS does not consider truncation to be a conversion error.

**Additional information**

The following additional information applies to the rounding function:

- For DECFLOAT columns, LOADPLUS always rounds to fit into the target column. The value of the DECFLOAT_ROUND_MODE option determines the rounding method.
- LOADPLUS performs any rounding before the scale function and before performing the REMOVEDECIMAL function.

**Converting to character data**

This section describes how LOADPLUS converts numeric data to character data.

*INTEGER EXTERNAL, BIGINT, INTEGER, and SMALLINT*

When converting an integer field to a CHAR column, LOADPLUS places your data in the CHAR column such that the last character of your input data is in the position described in the following table:

<table>
<thead>
<tr>
<th>Input</th>
<th>CHAR column length definition</th>
<th>Position of last character in CHAR column</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER EXTERNAL or BIGINT</td>
<td>Greater than or equal to 20</td>
<td>20th position</td>
</tr>
<tr>
<td></td>
<td>Less than 20</td>
<td>Last position</td>
</tr>
<tr>
<td>INTEGER or SMALLINT</td>
<td>Greater than or equal to 11</td>
<td>11th position</td>
</tr>
<tr>
<td></td>
<td>Less than 11</td>
<td>Last position</td>
</tr>
</tbody>
</table>

The following data conversion results occur:

- When the length of the value of your input data is smaller than the length of your CHAR column, LOADPLUS pads your column with leading blanks.
padding with leading blanks, LOADPLUS removes any leading zeros and, if applicable based on your field specification, any sign.

- When the length of your CHAR column is greater than the position of the last character of input data (as shown in Table 83 on page 380), LOADPLUS inserts blanks in the remainder of the column. For example, for BIGINT input, when the length of your CHAR column is greater than 20, LOADPLUS inserts blanks in positions 21 and greater.

- When the length of your CHAR column is smaller than the length of the value of your input data, LOADPLUS discards the record.

The following table illustrates how LOADPLUS handles integer input conversion to CHAR columns:

### Table 84: Sample INTEGER EXTERNAL, INTEGER, and SMALLINT to CHAR data conversions

<table>
<thead>
<tr>
<th>Data type input</th>
<th>Data type column</th>
<th>Data input a</th>
<th>Data column b</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER EXTERNAL(6)</td>
<td>CHAR(22)</td>
<td>'bbb100'</td>
<td>'bbbbbbbbbbbbbbbbbbbbbb100bb'</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>CHAR(10)</td>
<td>X'00000064'</td>
<td>'bbbbbb100'</td>
<td>Input value = 100</td>
</tr>
<tr>
<td>INTEGER</td>
<td>CHAR(14)</td>
<td>X'00000064'</td>
<td>'bbbbbbbb100bb'</td>
<td>Input value = 100</td>
</tr>
<tr>
<td>INTEGER</td>
<td>CHAR(3)</td>
<td>X'000003E8'</td>
<td>Error</td>
<td>Input value = 1000 Target too small Record discarded</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>CHAR(6)</td>
<td>X'03E8'</td>
<td>'bb1000'</td>
<td>Input value = 1000</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>CHAR(8)</td>
<td>X'83E8'</td>
<td>'bbb-1000'</td>
<td>Input value = -1000</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>CHAR(4)</td>
<td>X'83E8'</td>
<td>Error</td>
<td>Input value = -1000 Target too small Record discarded</td>
</tr>
<tr>
<td>SMALLINT REMOVESIGN</td>
<td>CHAR(4)</td>
<td>X'83E8'</td>
<td>'1000'</td>
<td>Input value = -1000</td>
</tr>
<tr>
<td>SMALLINT REMOVESIGN</td>
<td>CHAR(8)</td>
<td>X'83E8'</td>
<td>'bbbb1000'</td>
<td>Input value = -1000</td>
</tr>
</tbody>
</table>

a Integer input is shown in hexadecimal format.

b The letter b represents a blank.

---

**DECIMAL**

Chapter 3 Syntax of the LOAD command  381
When converting a DECIMAL field to a CHAR column, LOADPLUS performs the following conversion tasks:

- Unless your field specification includes the REMOVEDECIMAL option, LOADPLUS includes a place for the decimal point in the output column, regardless of whether you specify a value for scale in your input field.

- When the length of the value of your input data is smaller than the length of your CHAR column, LOADPLUS pads your column with leading blanks. Before padding with leading blanks, LOADPLUS removes any leading zeros and, if applicable based on your field specification, performs any REMOVESIGN and REMOVEDECIMAL functions.

- When the length of your CHAR column is smaller than the length of the value of your input data, LOADPLUS discards the record.

The following table illustrates how LOADPLUS handles DECIMAL input to CHAR columns:

**Table 85: Sample DECIMAL to CHAR data conversions**

<table>
<thead>
<tr>
<th>Data type input</th>
<th>Data type column</th>
<th>Data input a</th>
<th>Data column b</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECIMAL(5,0)</td>
<td>CHAR(6)</td>
<td>X'12345C'</td>
<td>'12345.'</td>
<td></td>
</tr>
<tr>
<td>DECIMAL(5,0)</td>
<td>CHAR(5)</td>
<td>X'12345C'</td>
<td>Error</td>
<td>Target too small Record discarded</td>
</tr>
<tr>
<td>DECIMAL(5,0)</td>
<td>CHAR(5)</td>
<td>X'02345C'</td>
<td>'2345.'</td>
<td></td>
</tr>
<tr>
<td>DECIMAL(5,0)</td>
<td>CHAR(6)</td>
<td>X'02345C'</td>
<td>'b2345.'</td>
<td></td>
</tr>
<tr>
<td>DECIMAL(5,0)</td>
<td>CHAR(6)</td>
<td>X'2345D'</td>
<td>'-2345.'</td>
<td></td>
</tr>
<tr>
<td>DECIMAL(5,0)</td>
<td>CHAR(5)</td>
<td>X'2345D'</td>
<td>Error</td>
<td>Target too small Record discarded</td>
</tr>
<tr>
<td>DECIMAL(5,0)</td>
<td>CHAR(5)</td>
<td>X'2345D'</td>
<td>'b2345'</td>
<td></td>
</tr>
</tbody>
</table>

a  Decimal input is shown in hexadecimal format.
b  The letter b represents a blank.

**Additional date time and timestamp fields considerations**

The following information applies to date, time, and timestamp fields:
A date, time, or timestamp EXTERNAL field consists of a valid date, time, or timestamp EBCDIC string as defined by DB2 SQL. LOADPLUS allows only trailing blanks.

For date, time, and timestamp fields, LOADPLUS loads the current date, time, or timestamp when the input data equates to spaces or zeros. However, for EXTERNAL date, time, and timestamp fields, if the column has a default value of null, LOADPLUS loads nulls when the input data equates to spaces or zeros.

The following information applies to field specifications for date, time, and timestamp fields:

- If you do not include any field specifications for a load job, LOADPLUS expects date, time, and timestamp data in external format.
- If you include a field specification for a date, time, or timestamp field, but do not specify the data type, LOADPLUS expects the data in external format.
- If you include a field specification for a date, time, or timestamp field, but do not specify a data format, LOADPLUS expects the data in internal format.

**Additional VARCHAR VARGRAPHIC VARBINARY ROWID and XML fields considerations**

For these data types, a two-byte length precedes the character, graphic, or hexadecimal data.

The length does not include the two length bytes. For VARCHAR, VARBINARY, ROWID, and XML, the length is the number of bytes. For VARGRAPHIC, the length is the number of double-byte characters.

**Data translation**

Before writing the input record to the table, LOADPLUS translates the data from the specified encoding scheme of the input data.

LOADPLUS determines the encoding scheme based on the values of the EBCDIC, ASCII, or UNICODE command option and the CCSID command option.

**Translation processing**

LOADPLUS handles translation processing in the follow manner:
1 Based on the CCSID that you specify or that LOADPLUS retrieves from DSNHDECP, LOADPLUS searches the SYSIBM.SYSSTRINGS catalog table for a row that identifies how to translate the data. (To review the values that are assigned to the CCSID, see the LOADPLUS parameter listing under message number BMC50471I.)

2 If a row exists, LOADPLUS searches the row for a value, first in the TRANSPROC field and then in the TRANSTAB field. LOADPLUS invokes DSNUTILB when either of the following conditions exists:
   - The TRANSPROC field contains DSNXVJPC or DSNXVTWC.
   - The TRANSTAB field is empty, and the TRANSPROC field contains a value.

In all other cases, LOADPLUS performs the data translation providing the translation is supported. For a list of supported translations, see Table 86 on page 384.

3 If a row does not exist in SYSIBM.SYSSTRINGS, LOADPLUS passes the data to the z/OS Unicode Conversion Services for translation.

**Supported character conversions**

The following table identifies the character conversions that LOADPLUS supports for translation processing. For more restrictions on translation processing, see “Restrictions” on page 385.

This table indicates the supported translations with the following identifiers:

- **X** = allowable translation
- **D** = allowable translation by invoking DSNUTILB

**Table 86: Supported translations**

<table>
<thead>
<tr>
<th>Target</th>
<th>ASCII SBCS</th>
<th>ASCII MIXED</th>
<th>ASCII DBCS</th>
<th>EBCDIC SBCS</th>
<th>EBCDIC MIXED</th>
<th>EBCDIC DBCS</th>
<th>UNICODE SBCS</th>
<th>UNICODE MIXED</th>
<th>UNICODE DBCS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII SBCS</td>
<td>X</td>
<td>X, D&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
<td>D</td>
<td></td>
<td>X</td>
<td></td>
<td>X, D&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>ASCII MIXED</td>
<td>D</td>
<td>X, D&lt;sup&gt;b&lt;/sup&gt;</td>
<td>D</td>
<td>D</td>
<td></td>
<td>D</td>
<td></td>
<td>D, D</td>
<td></td>
</tr>
<tr>
<td>ASCII DBCS</td>
<td></td>
<td></td>
<td>X</td>
<td>D&lt;sup&gt;a&lt;/sup&gt;</td>
<td></td>
<td></td>
<td></td>
<td>D&lt;sup&gt;b&lt;/sup&gt;</td>
<td></td>
</tr>
<tr>
<td>EBCDIC SBCS</td>
<td>X</td>
<td>D</td>
<td>X</td>
<td>X, D&lt;sup&gt;a&lt;/sup&gt;</td>
<td></td>
<td>X</td>
<td></td>
<td>X, D&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>EBCDIC MIXED</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>X, D&lt;sup&gt;b&lt;/sup&gt;</td>
<td></td>
<td>D</td>
<td></td>
<td>D, D</td>
<td></td>
</tr>
</tbody>
</table>
The translation process might cause the length of the resultant data to change based on the CCSID type that you specified. To determine the resultant length of the data based on each CCSID conversion, see the IBM DB2 SQL Reference.

If an expansion error occurs during translation, LOADPLUS writes the record to the discard data set (SYSDISC).

**Translations that invoke DSNUTILB**

If you request a translation that requires LOADPLUS to invoke DSNUTILB, ensure that you supply the correct options and environment for invoking DSNUTILB. Otherwise, LOADPLUS issues BMC50178E and terminates.

- For the requirements for a basic DSNUTILB load job, see “Load jobs that invoke DSNUTILB” on page 57.
- In addition, the UNICODE installation option must be set to YES. This value must be in effect for any translation job that invokes DSNUTILB, regardless of whether you are loading Unicode data.

**Restrictions**

The following table describes the restrictions on data translation for LOADPLUS:

<table>
<thead>
<tr>
<th>Source</th>
<th>ASCII SBCS</th>
<th>ASCII MIXED</th>
<th>ASCII DBCS</th>
<th>EBCDIC SBCS</th>
<th>EBCDIC MIXED</th>
<th>EBCDIC DBCS</th>
<th>UNICODE SBCS</th>
<th>UNICODE MIXED</th>
<th>UNICODE DBCS</th>
</tr>
</thead>
<tbody>
<tr>
<td>EBCDIC DBCS</td>
<td>D</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D</td>
</tr>
<tr>
<td>UNICODE SBCS</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UNICODE MIXED</td>
<td>X&lt;sup&gt;c&lt;/sup&gt;</td>
<td>D</td>
<td>X&lt;sup&gt;c&lt;/sup&gt;</td>
<td>D</td>
<td>X&lt;sup&gt;c&lt;/sup&gt;</td>
<td>X&lt;sup&gt;c&lt;/sup&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UNICODE DBCS</td>
<td>D</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

* a If the CCSIDs of the input do not match the CCSIDs of the DB2 subsystem, LOADPLUS invokes DSNUTILB.
* b If you are loading data from a referenced file, LOADPLUS invokes DSNUTILB.
* c If the input value is greater than the column length, LOADPLUS trims any trailing blanks to fit.
Table 87: Restrictions on data translation

<table>
<thead>
<tr>
<th>Conditions</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Your job uses both of the following specifications:</td>
<td>LOADPLUS ignores the data translation options.</td>
</tr>
<tr>
<td>■ FORMAT BMCUNLOAD</td>
<td></td>
</tr>
<tr>
<td>■ One or more data translation options, such as CCSID.</td>
<td></td>
</tr>
<tr>
<td>Columns have one of the following definitions:</td>
<td>LOADPLUS does not translate these columns.</td>
</tr>
<tr>
<td>■ The FOR BIT DATA attribute</td>
<td></td>
</tr>
<tr>
<td>■ The BINARY or VARBINARY data type</td>
<td></td>
</tr>
</tbody>
</table>

**Constants**

For comparisons between predicate block character constants and input data, LOADPLUS translates constants from EBCDIC (using the EBCDIC SBCS system default CCSID) to the encoding scheme of the input data.

For constants that are placed in the table data, LOADPLUS translates the constants from EBCDIC (using the EBCDIC SBCS system default CCSID) to the table encoding scheme. LOADPLUS translates the following character constants for loading:

- VALUE constants
- DEFAULTIF VALUE constants

**SUBBYTE and ERRORBYTE fields**

LOADPLUS uses SUBBYTE and ERRORBYTE fields in the same way that the IBM DB2 LOAD utility uses them.

*Note*

With the DB2 LOAD utility, if you specify NOSUBS and discard processing is not active, the job fails when it encounters records that require substitution. LOADPLUS does not fail under these circumstances. However, LOADPLUS processes only the records that do not require substitution.

**Order of data type conversion and data translation**

The following tables detail when data translation occurs with respect to data type conversion. The character combinations in the tables indicate the order in which data type conversion, data translation, and character padding occur during data translation:

- x indicates supported conversion, no translation required
- c indicates conversion
- t indicates translation
- p indicates padding with blanks

The numbers represent the type of data translation that occurs during data type conversion:

- 1 indicates translation of the input CCSID to EBCDIC SBCS (system default)
- 2 indicates translation of EBCDIC SBCS (system default) to output CCSID
- 3 indicates translation of input CCSID to output CCSID

### Table 88: Conversion and translation processing for numeric output columns

<table>
<thead>
<tr>
<th>Output column</th>
<th>SMALLINT</th>
<th>INTEGER</th>
<th>BIGINT</th>
<th>DECIMAL</th>
<th>FLOAT</th>
<th>DECFLOAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>INTEGER</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>INTEGER</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
</tr>
<tr>
<td>EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
</tr>
<tr>
<td>EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZONED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>FLOAT</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
</tr>
<tr>
<td>EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output column</td>
<td>SMALLINT</td>
<td>INTEGER</td>
<td>BIGINT</td>
<td>DECIMAL</td>
<td>FLOAT</td>
<td>DECFLOAT</td>
</tr>
<tr>
<td>------------------</td>
<td>----------</td>
<td>---------</td>
<td>--------</td>
<td>---------</td>
<td>-------</td>
<td>----------</td>
</tr>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE-format</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-format</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP-format</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XML</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXIT</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

Table 89: Conversion and translation processing for binary, character, and graphic output columns

<table>
<thead>
<tr>
<th>Output column</th>
<th>BINARY</th>
<th>VARBINARY</th>
<th>CHARACTER</th>
<th>VARCHAR</th>
<th>GRAPHIC</th>
<th>VARGRAPHIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td></td>
<td>cpt 2</td>
<td>cpt 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td></td>
<td></td>
<td>cpt 2</td>
<td>cpt 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER EXTERNAL</td>
<td></td>
<td></td>
<td>tcpt 1,2</td>
<td>tcpt 1,2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output column</td>
<td>BINARY</td>
<td>VARBINARY</td>
<td>CHARACTER</td>
<td>VARCHAR</td>
<td>GRAPHIC</td>
<td>VARGRAPHIC</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------</td>
<td>-----------</td>
<td>-----------</td>
<td>---------</td>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td></td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td></td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td></td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td></td>
<td>cpt 3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td>cpt 3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>GRAPHIC EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE-format EXTERNAL</td>
<td></td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td>ctp 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-format EXTERNAL</td>
<td></td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td>ctp 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP-format EXTERNAL</td>
<td></td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
<td>tcpt 1,2</td>
<td></td>
</tr>
</tbody>
</table>
### Table 90: Conversion and translation processing for date, time, and timestamp output columns

<table>
<thead>
<tr>
<th>Output column</th>
<th>DATE</th>
<th>TIME</th>
<th>TIMESTAMP</th>
<th>TIMESTAMP WITH TIME ZONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER EXTERNAL</td>
<td>tc 1</td>
<td>tc 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td>tc 1</td>
<td>tc 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td>tc 1</td>
<td>tc 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output column</td>
<td>DATE</td>
<td>TIME</td>
<td>TIMESTAMP</td>
<td>TIMESTAMP WITH TIME ZONE</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------</td>
<td>------</td>
<td>-----------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td>tc 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE-format EXTERNAL</td>
<td>tc 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td>tc 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-format EXTERNAL</td>
<td>tc 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP-format EXTERNAL</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE EXTERNAL</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
<td>tc 1</td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XML</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXIT</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

Table 91: Conversion and translation processing for LOB and XML output columns

<table>
<thead>
<tr>
<th>Output column</th>
<th>BLOB</th>
<th>CLOB</th>
<th>DBCLOB</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output column</td>
<td>BLOB</td>
<td>CLOB</td>
<td>DBCLOB</td>
<td>XML</td>
</tr>
<tr>
<td>---------------</td>
<td>------</td>
<td>------</td>
<td>--------</td>
<td>-----</td>
</tr>
<tr>
<td><strong>Input field</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE-format EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME-format EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP- format EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE EXTERNAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td><strong>x</strong></td>
</tr>
</tbody>
</table>
Dynamic allocation options

Dynamic allocation options allow you to have LOADPLUS optimally allocate the size and number of work files and copy data sets that are needed for your load job.

*Note*
LOADPLUS does not use these options to dynamically allocate input (SYSREC) data sets. See “INDSN” on page 211 for information about dynamically allocating input data sets.

Dynamic allocation provides the following benefits:

- Eliminates the need to include DD statements for the specified file types in your JCL

  *Note*
  If you activate dynamic allocation *and* you specify DD statements in your JCL, use the IFALLOC installation or command option to control whether LOADPLUS dynamically allocates your data sets or uses the data sets that you allocated in your JCL.

- Eliminates the need to calculate data set allocation sizes

- Automatically provides an optimal allocation for the highest possible multitasking level

- Allows you to use symbolic variables and user-defined variables to assist in creating data set names

The following sections provide the syntax that is required for enabling and using dynamic allocation. For more information, see “Dynamic data set allocation in LOADPLUS” on page 78.
DSNUTILB load jobs

For a DSNUTILB load, you must enable dynamic allocation for all work files that the load job requires by specifying ACTIVE YES for those DDTYPEs. For copy data sets, ACTIVE YES is required for at least the primary local copy data set (DDTYPE LOCPFCPY) if you specify COPY YES. LOADPLUS ignores any copy or work file data set allocation in your JCL.

**Note**

All copy data sets for a DSNUTILB load are dynamically allocated, even if you specify ACTIVE YES for only the primary local copy data set.

For these jobs, LOADPLUS uses your dynamic allocation options to generate a TEMPLATE control statement for each data set. LOADPLUS then passes these TEMPLATE statements to DSNUTILB for data set allocation. For more information about TEMPLATE control statements, see the documentation for the IBM DB2 LOAD utility.

**AUTOENUMROWS and ENUMROWS options**

Unless you are running a load job that invokes DSNUTILB, dynamic allocation requires one of the following specifications:

- Enable automated file size estimation by either specifying or defaulting to AUTOENUMROWS=YES in your installation options, or specifying ENUMROWS AUTO on your LOAD command.

- Specify an appropriate numeric value for the ENUMROWS option.

For more information, see “ENUMROWS” on page 229.

**MAXTAPE**

The MAXTAPE option allows you to specify the maximum number of tape devices that LOADPLUS can use when dynamically allocating data sets.

For details regarding how LOADPLUS handles dynamic data set allocation when it reaches your MAXTAPE limit, see “Reaching the MAXTAPE limit during dynamic allocation” on page 84.
If you are loading a partition-by-growth table space and you specify COPYLVL
PART to take partition-level copies, use the following formula to determine the copy
portion of the MAXTAPE value. Add this value to any tapes that you need for
dynamically allocated work files.

\[(\text{# of existing partitions} + 1) \times \text{# of copies per partition}\]

**Restriction**

When invoking DSNUTILB, LOADPLUS ignores this option.

**Specifying the default**

You can specify the default for the MAXTAPE command option in your installation
options module by using the MAXTAPE installation option. LOADPLUS was
shipped with a default value of 3 for this option. The command option overrides the
default that is in the installation options module.

**DSNUEXIT**

The DSNUEXIT option allows you to specify a user-written exit routine that creates
user-defined variables.

You can use these variables with the DSNPAT option to create patterns for data set
names for dynamically allocated data sets. For details about the DSNPAT option, see
“DSNPAT” on page 414.

**Restriction**

When invoking DSNUTILB, LOADPLUS ignores this option.

**Specifying the default**

You can also specify a default for the DSNUEXIT option in your installation options
module by using the DSNUEXIT installation option. LOADPLUS was shipped with
a default value of (NONE,ASM) for this option. The DSNUEXIT command option overrides any default that is in the installation options module.

**exitName**

Specify the name of the exit routine. This exit routine must be authorized and present in your system’s LINKLIST or your JOBLIB or STEPLIB. For COBOL II and C programs, the appropriate runtime libraries must be authorized and in your LINKLIST, JOBLIB, or STEPLIB. For the IBM Language Environment COBOL and C programs (LE COBOL and LE C), the appropriate language environment runtime libraries must be authorized and in your LINKLIST, JOBLIB, or STEPLIB.

For more information about creating an exit routine, see “LOADPLUS user exits” on page 825.

**NONE**

Specify NONE in place of an exit name to override a default user exit in your installation options. This tells LOADPLUS that you do not want to call any DSNUEXIT user exit.

**program language**

LOADPLUS supports user-written exit routines written in assembler, COBOL II, LE COBOL, C, or LE C. After you specify the name of your user exit, you can optionally provide the program language in which it was written. If you specify the program language, place a comma between the exit name and the program language. The following table lists the keywords that you can use to specify a program language.

**Note**

The program language, whether specified here or in the installation option, must match the source language of the specified user exit. If the language does not match, your user exit might not function correctly.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASM</td>
<td>Specifies an exit written in assembler</td>
</tr>
<tr>
<td>COBOL2</td>
<td>Specifies an exit written in COBOL II</td>
</tr>
<tr>
<td>LE_COBOL</td>
<td>Specifies an exit written in LE COBOL</td>
</tr>
<tr>
<td>C</td>
<td>Specifies an exit written in C</td>
</tr>
<tr>
<td>LE_C</td>
<td>Specifies an exit written in LE C</td>
</tr>
</tbody>
</table>
DDTYPE

The DDTYPE option tells LOADPLUS to dynamically allocate one or more types of data sets or to override an active dynamic allocation value in your installation options.

Additional options in the DD type block provide control for various aspects of the dynamic allocation process. You can repeat this DD type block for more than one data set type. The following table lists the keywords that you can specify for the data set types that you want to allocate dynamically:

Table 93: DDTYPE keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD</td>
<td>Load data files (SORTOUT)</td>
</tr>
<tr>
<td>WORK</td>
<td>Index work files (SYSUT1)</td>
</tr>
</tbody>
</table>
### Specifying the default

You can also specify the data set type in your installation options module by using the DDTYPE installation option. Any dynamic data set allocation options that you specify with the DDTYPE command option override the defaults that are in the installation options module for the same DDTYPE.

### ACTIVE

The ACTIVE option allows you to specify whether you want dynamic allocation active for the specified data set type.

![ACTIVE diagram]

If you specify the ACTIVE keyword without a value, LOADPLUS assumes ACTIVE YES.

You cannot change the value of the ACTIVE option when restarting your load job.
DSNUTILB load jobs

For a DSNUTILB load, you must specify ACTIVE YES on the DDTYPEs for all work files that the load job requires. If you also specify COPY YES, you must specify ACTIVE YES for at least the local primary copy data set (DDTYPE LOCPFCPY). LOADPLUS ignores any copy or work file data set allocation in your JCL.

Note

All copy data sets for a DSNUTILB load are dynamically allocated, even if you specify ACTIVE YES for only the local primary copy data set.

LOB and XML load jobs

If you specify COPY YES when loading LOB or XML data, you must enable dynamic allocation for at least the primary local copy data set (DDTYPE LOCPFCPY or DDTYPE LOCPXCPY). In general, LOADPLUS ignores any LOB or XML copy data set allocation in your JCL. However, if you specify any DD statements in your JCL that use one of the default LOB or XML copy prefixes, LOADPLUS terminates.

Specifying the default

You can specify the default for the ACTIVE command option in your installation options module by using the ACTIVE installation option. For most DDTYPEs, LOADPLUS was shipped with a default value of YES for this option. The command option overrides the default that is in the installation options module.

When running in a CHANGE MANAGER worklist environment, LOADPLUS ignores the ACTIVE option in your installation options module. LOADPLUS dynamically allocates your data sets only if CHANGE MANAGER supplies the ACTIVE YES syntax.

YES

This option tells LOADPLUS to activate dynamic allocation for the specified data set type. ACTIVE YES must be in effect for LOADPLUS to use the remaining dynamic allocation options.

Note

When determining whether to actually dynamically allocate a data set, LOADPLUS takes into account the values of both the ACTIVE and IFALLOC options for that data set type.

NO

This option tells LOADPLUS to inactivate dynamic allocation for the specified data set type. Use this option if dynamic allocation is set to active in your installation.
options but you do not want LOADPLUS to dynamically allocate the specified data sets for a particular processing session.

**IFALLOC**

The IFALLOC option tells LOADPLUS to either free the data sets specified in your JCL or use them and, if necessary, allocate additional data sets.

**Restrictions**

LOADPLUS ignores this option in the following cases:

- When invoking DSNUTILB
- For LOB and XML copy data sets

**Specifying the default**

You can specify the default for the IFALLOC command option in your installation options by using the IFALLOC installation option. LOADPLUS was shipped with a default value of USE for this option. The command option overrides the default that is in the installation options module.

**USE**

Specifying this option tells LOADPLUS to use the data sets that you allocated in your JCL. If the number of data sets that you specify in your JCL is insufficient for processing, LOADPLUS (or BMCSORT in the case of sort work files) dynamically allocates the additional files that your job needs. LOADPLUS uses both the dynamically allocated data sets and those that you specify in your JCL. LOADPLUS allocates only additional files, not additional space for a single file.

When you specify IFALLOC USE, the ddnames that you specify in the JCL must be eight bytes long and must match the corresponding ddnames displayed in one of the following reports in the SYSPRINT data set:

- LOADPLUS DASD REQUIREMENT ESTIMATES (messages BMC51530I through BMC51533I)

  LOADPLUS issues this report when you specify ANALYZE PAUSE or ANALYZE ONLY.
DYNAMIC FILE ALLOCATION REPORT (messages BMC50445I through BMC50448I)
LOADPLUS issues this report when dynamic data set allocation is active.

FREE

This option tells LOADPLUS to free the data sets that you allocated in your JCL and use only dynamically allocated data sets.

Note
If you specify this option for a load job in a worklist environment, the data sets are freed for the remainder of the worklist execution. This can result in an error if a subsequent LOADPLUS job in the worklist requires these data sets.

SMS

The SMS specifies whether to pass the SMS classes and extended data set types (EXTREQ and EXTPREF) to SMS during dynamic allocation.

Whether the utility actually performs an SMS allocation depends on your site.

Specifying the default

You can specify the default for the SMS command option in your installation options module by using the SMS installation option. The utility was shipped with a default value of NO for this option. The command option overrides the default that is in the installation options module.

YES

YES passes SMS classes and extended data set types during dynamic allocation. The following considerations apply to SMS YES:

- When you specify SMS YES, the utility ignores the value that you specify for MAXEXTSZ.
- To pass the UNIT value to SMS during dynamic allocation, also specify the SMSUNIT option.
When invoking DSNUTILB, the utility includes the SMS classes in the TEMPLATE control statement that it builds for the IBM utility.

**NO**

NO does not pass SMS classes and extended data set types during dynamic allocation. When you specify SMS NO, the utility always passes the UNIT value during dynamic allocation.

**SMSUNIT**

The SMSUNIT option specifies whether to pass the UNIT value in the SMS allocation parameter list to SMS during dynamic allocation.

If you do not specify SMS YES, the utility ignores the SMSUNIT option. The utility does not modify any other parameters based on this option.

**Specifying the default**

You can specify the default for the SMSUNIT command option in your installation options module by using the SMSUNIT installation option. The utility was shipped with a default value of NO for this option. The command option overrides the default that is in the installation options module.

**NO**

NO does not pass the value of the UNIT option.

**YES**

YES passes the value of the UNIT option.

When invoking DSNUTILB, the utility includes the UNIT option values in the TEMPLATE control statements that it builds for the IBM utility.

**MAXEXTSZ**

For any extent that LOADPLUS allocates for a dynamically allocated data set, the MAXEXTSZ option allows you to specify the maximum allowable value for the primary space allocation.
Because the secondary quantity cannot exceed the primary quantity, MAXEXTSZ also controls the maximum secondary quantity.

Specify MAXEXTSZ by using one of the following values:

- 0 if you do not want to set a limit for space allocation
- An integer for the units specified with the UNIT option, with the unit of measure as follows:
  - K for kilobytes
  - TRK for tracks
  - CYL for cylinders

You can specify values for MAXEXTSZ by using one of the following options:

- If you specify a single value (integer1), that value applies to both units.
- If you specify a single value and a comma as follows, LOADPLUS takes the missing value from the MAXEXTSZ installation option:
  \[ \text{MAXEXTSZ} \left( \left( \text{integer1}, \text{K} \right) \right) \]
  or
  \[ \text{MAXEXTSZ} \left( \left( \text{integer2}, \text{K} \right) \right) \]
- If you specify a second value (integer2), LOADPLUS uses that value when the value specified for the THRESHLD option is exceeded. This value applies to the secondary device type that you specify with the UNIT option (or, if applicable, through SMS).

Specifying two MAXEXTSZ values might be useful if characteristics of the unit that you specify for allocations below the threshold are different from those of the unit that you specify for allocations above the threshold.

Regardless of the amount of space that LOADPLUS determines that it needs, LOADPLUS will not allocate more than your specified MAXEXTSZ limit for either the primary or the secondary quantity:
If the amount of required space that LOADPLUS calculates is greater than the MAXEXTSZ limit for the primary quantity, LOADPLUS uses the secondary extents to hold the remainder of the required primary space.

If the amount of required space that LOADPLUS calculates cannot be accommodated because of MAXEXTSZ restrictions, the job might terminate with an out-of-space condition on the data set.

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS includes the first parameter of this option as the MAXPRIME value in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. For DSNUTILB, the unit of measure is always cylinders.

If building a secondary TEMPLATE control statement for your copy data sets, LOADPLUS includes the second parameter of this option as the MAXPRIME value in that template. For information about when LOADPLUS builds a secondary template, see “THRESHLD” on page 411.

**Additional considerations**

Note the following additional information about the MAXEXTSZ option:

- The MAXEXTSZ option is not valid for DDTYPE SORTWORK.
- LOADPLUS ignores MAXEXTSZ when you specify SMS YES.
- LOADPLUS checks the value of MAXEXTSZ after applying SIZEPCT to the allocation amount.

**Specifying the default**

You can specify the default for the MAXEXTSZ command option in your installation options module by using the MAXEXTSZ installation option. LOADPLUS was shipped with a default value of 0 for this option. The command option overrides the default that is in the installation options module.

**DSNTYPE**

The DSNTYPE option enables you to specify the type of data set that you want to allocate.
If you specify a second value, the utility uses that value when the value that is in effect for the THRESHLD option is exceeded.

**Restrictions**

The following restrictions apply to the DSNTYPE option:

- The DSNTYPE option is not valid for tape data sets.
- When invoking DSNUTILB, the utility ignores this option.

**Specifying the default**

You can specify the default for the DSNTYPE command option in your installation options module by using the DSNTYPE installation option. The utility was shipped with a default value of (NONE,NONE) for this option. The command option overrides the default that is in the installation options module.

You can specify one of the following options:

**Table 94: DSNTYPE option keywords**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>Specifies not use any extended attributes for this data set allocation</td>
</tr>
<tr>
<td>LARGE</td>
<td>Allocates this data set as a large format sequential data set. This option enables data sets larger than 65,535 tracks.</td>
</tr>
<tr>
<td>BASIC</td>
<td>Allocates this data set as a basic sequential data set. This data set will be limited to 65,535 tracks.</td>
</tr>
<tr>
<td>EXTREQ</td>
<td>Allocates this data set as an extended format data set. This option is ignored if either of the following conditions exists: The SMS option is NO. The DDTYPE is SORTWORK.</td>
</tr>
</tbody>
</table>


**EXTPREF**

Specifies that you prefer that this data set be allocated as an extended format data set. If this allocation is not possible, the data set is allocated as a basic format data set.

This option is ignored if either of the following conditions exists:
- The SMS option is NO.
- The DDTYPE is SORTWORK.

**UNIT**

The UNIT option allows you specify unit names for use during dynamic allocation.

```
UNIT ( SYSALLDA unitName1, SYSALLDA unitName2 )
```

How the utility uses these values depends on the type of data sets that you are allocating:

- For non-SMS-managed data sets, the UNIT option allows you to specify the primary and secondary unit names to use for dynamic data set allocation. The unit names cannot exceed eight characters. Enclose the two unit names within parentheses and separate them with a comma.

- For SMS-managed data sets when you specify SMS YES and SMSUNIT YES, this option supplies the unit names to pass in the SMS allocation parameter list. DSNUTILB reorganization jobs function differently, as described in the DSNUTILB reorganization jobs section.

The THRESHLD installation or command option controls which of the specified unit names the utility selects.

**DSNUTILB reorganization jobs**

When invoking DSNUTILB, the utility includes the first parameter in the TEMPLATE control statement that it builds for the IBM utility.

If building a secondary TEMPLATE control statement for your copy data sets, the utility includes the second UNIT parameter in that template. For information about building a secondary template, see the description of the THRESHOLD option.

**Specifying the default**
You can specify the default for the UNIT command option in your installation options module by using the UNIT installation option. The utility was shipped with a default value of (SYSALLDA, SYSALLDA) for this option. The command option overrides the default that is in the installation options module.

**UNITCNT**

This option enables you to specify the number of devices to allocate when dynamically allocating data sets.

\[
\text{UNITCNT} \rightarrow (\text{integer1}, 0, \text{integer2})
\]

Valid values are 0 through 59. A value of 0 tells the utility to use the system default.

If you specify a second value (\text{integer2}), the utility uses this number when the value for THRESHLD is exceeded. This value applies to the secondary device type that you specify with the UNIT option (or, if applicable, through SMS).

**DSNUTILB reorganization jobs**

When invoking DSNUTILB, the utility includes the first parameter in the TEMPLATE control statement that it builds for the IBM utility.

If building a secondary TEMPLATE control statement for your copy data sets, the utility includes the second UNITCNT parameter in that template. For information about building a secondary template, see the description of the THRESHLD option.

**Additional considerations**

The following additional considerations apply to the UNITCNT option:

- The utility ignores this option for sort work files.
- To avoid performance problems, specify only the number of devices that you need.

**Specifying the default**

You can specify the default for the UNITCNT command option in your installation options module by using the UNITCNT installation option. The utility was shipped with a default value of 0 for this option. The command option overrides the default that is in the installation options module.
VOLCNT

This option enables you to specify the maximum number of volumes to use during dynamic allocation.

Valid values for this option are as follows:

- 0 to not specify a volume count for dynamic allocation.
- Integer values 1 through 255 to specify the number of volumes.
- AUTO to compute the volume count based on the amount of data, adjusted for the estimated space required (based on, for example, the value of the SIZEPCT installation or command option).

The utility computes the volume count by dividing the size estimate by the value specified for the AVGVOLSP installation or command option.

If you specify a second value (integer2), the utility uses that value when the value for the THRESHLD installation or command option is exceeded. This value applies to the secondary device type that you specify with the UNIT option (or, if applicable, through SMS).

**DSNUTILB reorganization jobs**

When invoking DSNUTILB, the utility includes the first parameter in the TEMPLATE control statement that it builds for the IBM utility.

If building a secondary TEMPLATE control statement for your copy data sets, the utility includes the second VOLCNT parameter in that template. For information about building a secondary template, see the description for the THRESHLD option.

**Additional considerations**

Note the following additional information about the VOLCNT option:

- This option is not available for DDTYPE SORTWORK. If you specify a value for this option for sort work files, the value is changed to 1.
- To avoid performance problems, specify only the number of volumes that you need.
For an SMS-managed data set, BMC recommends that you specify a value of 0 if your ACS routines are set up to provide a volume count.

**Specifying the default**

You can specify the default for the VOLCNT command option in your installation options module by using the VOLCNT installation option. The utility was shipped with a default value of 25 for this option. The command option overrides the default that is in the installation options module.

**AVGVOLSP**

The AVGVOLSP option enables you to specify the average amount of space that is available on each device.

Use this option to reflect the average space available on volumes that are eligible to contain the dynamically allocated data set.

Specify one or two integer values, and include one of the following units of measure:

- K for kilobytes
- TRK for tracks (the default)
- CYL for cylinders

The utility uses AVGVOLSP only when you specify AUTO for the corresponding first or second parameter of the VOLCNT installation or command option.

If you specify a second value (\(integer_2\)), the utility uses that value when the value for the THRESHLD option is exceeded. This value applies to the secondary device type that you specify with the UNIT option (or, if applicable, through SMS).

**Restrictions**

The following restrictions apply to the AVGVOLSP option:

- The AVGVOLSP option is not valid for DDTYPE SORTWORK.
- When invoking DSNUTILB, the utility ignores this option.

**Additional considerations**
Note the following additional information about the AVGVOLSP option:

- You should not use AVGVOLSP to specify the maximum space on all devices or volumes unless the volumes to be used are empty.

- If you specify a value for AVGVOLSP that is too small, the utility computes a value for VOLCNT that is too large. However, if you specify a value for this option that is too large, the utility computes a value for VOLCNT that is too small.

**Specifying the default**

You can specify the default for the AVGVOLSP command option in your installation options module by using the AVGVOLSP installation option. The utility was shipped with a default value of ((30000,TRK),(30000,TRK)) for this option. The command option overrides the default that is in the installation options module.

**SIZEPCT**

The SIZEPCT option allows you to adjust, by percentages, the allocated data set sizes that LOADPLUS calculated.

The numbers that you specify must be greater than 0.

- **primary** indicates the percentage of the primary data set size calculated by LOADPLUS that you actually want allocated.

- **secondary** indicates the percentage of the secondary data set size calculated by LOADPLUS that you actually want allocated.

If you specify a secondary size that is greater than the primary size, LOADPLUS changes the secondary value to equal the primary value.

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS includes the first parameter of this option as the PCTPRIME value in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. If you specify a value greater than 100, LOADPLUS converts it to 100.

**Specifying the default**
You can specify the default for the SIZEPCT command option in your installation options module by using the SIZEPCT installation option. LOADPLUS was shipped with a default value of (100,100) for this option. The command option overrides the default that is in the installation options module.

**SPACE**

The SPACE option enables you to override the DSNUTILB space calculation for the dynamically allocated data set.

```
SPACE (primary,secondary) CYL TRK
```

*This option is meaningful only for a DSNUTILB reorganization.*

This option also overrides the value for the SIZEPCT option.

When you use this option, you must specify integer values for both the primary and secondary data sets. The numbers that you specify must be greater than 0. Include the unit of measure as follows:

- CYL for cylinders
- TRK for tracks

When not invoking DSNUTILB, the utility ignores this option.

**THRESHLD**

The THRESHLD option allows you to specify a threshold value, in kilobytes, above which the utility applies secondary values to allocated data sets.

```
THRESHLD 0 integer
```

The utility tests this threshold for each data set to be allocated. If the size for a particular data set is greater than the threshold, the utility performs the following tasks:

- When SMS is NO, the utility uses the secondary values of the following options for the allocated data sets:
  - UNIT (unit name)
- UNITCNT (unit count)
- VOLCNT (volume count)
- AVGVOLSP (average volume space)
- MAXEXTSZ (extent size)
- DSNTYPE (data set type), unless DSNTYPE is EXTREQ or EXTPREF

- When SMS is YES, the utility uses the secondary values of the following options (or passes them to SMS if applicable):
  - STORCLAS, MGMTCLAS, or DATACLAS (class name for the SMS classes)
  - VOLCNT (volume count)
  - AVGVOLSP (average volume space)
  - DSNTYPE (data set type)
  - If SMSUNIT is YES
    - UNIT (unit name)
    - UNITCNT (unit count)

If you specify 0, or if the threshold is not exceeded, the utility uses the primary values for these options.

**Tip**
If you use THRESHLD to send larger data sets to tape, consider setting the MAXTAPE option to limit the number of tapes that you use.

**DSNUTILB reorganization jobs**
When invoking DSNUTILB, the utility handles this option differently, depending on the type of data set that you are allocating:

- For copy data sets, the utility translates this option to a LIMIT value in the TEMPLATE control statement that it builds for the corresponding IBM utility. The utility also builds the secondary TEMPLATE control statement to which the IBM utility will switch when the LIMIT is exceeded.
  
The utility assumes that the value that you supply for THRESHLD is in kilobytes and translates that value to the appropriate value and unit of measure for the LIMIT keyword. Note the following additional information about this value:
  - The utility rounds down to the nearest whole value.
  - If you specify a value that would cause the utility to translate to a value less than 1 cylinder, the utility builds the template with a LIMIT value of 1 CYL.

- For all other data sets, the utility ignores this option.

**Examples**
The following examples illustrate how to use THRESHLD for different types of scenarios. In example 1, all allocated data sets for this DDTYPE go to SYSDA:

**Figure 22: THRESHLD example 1**

```
UNIT(SYSDA)
THRESHLD 0
```

In example 2, allocated data sets with a size greater than 720 MB for this DDTYPE go to unit LARGE. Smaller allocated data sets for this DDTYPE go to unit WORK:

**Figure 23: THRESHLD example 2**

```
UNIT(WORK,LARGE)
THRESHLD 720000
```

In example 3, data sets for this DDTYPE are SMS-managed. Allocated data sets with a size greater than 72 MB go to STORCLS2, MGMTCLS2, and DATACLS2. Smaller data sets go to the classes specified in the first parameter of each class type.

**Figure 24: THRESHLD example 3**

```
SMS YES
STORCLAS(STORCLS1,STORCLS2)
MGMTCLAS(MGMTCLS1,MGMTCLS2)
DATACLAS(DATACLS1,DATACLS2)
THRESHLD 720000
```

In example 4, the utility allocates data sets with the following properties for this DDTYPE:

- Data sets with a size less than or equal to 7,200 MB go to unit SYSDA. The maximum volume count is calculated based on an average volume usage of 30,000 tracks.
- Data sets with a size greater than 7,200 MB go to unit CART with no more than 25 volumes.

**Figure 25: THRESHLD example 4**

```
VOLCNT(AUTO,25)
AVGVOLSP((30000,TRK),(900000,TRK))
UNIT(SYSDA,CART)
THRESHLD 7200000
```

**Specifying the default**

You can specify the default for the THRESHLD command option in your installation options module by using the THRESHLD installation option. The utility was shipped with a default value of 0 for this option. The command option overrides the default that is in the installation options module.
DSNPAT

The DSNPAT option directs LOADPLUS to use a particular pattern of variables and text to create data set names for dynamic allocation.

The maximum total length that LOADPLUS allows for a data set name is 44 bytes. LOADPLUS removes any trailing blanks in the resolved pattern.

When invoking DSNUTILB, LOADPLUS includes this pattern in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. Any variables that you include in your pattern for this type of load must be either valid for the IBM DB2 TEMPLATE control statement or translatable (as shown in Table 95 on page 415) to a valid TEMPLATE variable. For more information about TEMPLATE control statements, see the documentation for the DB2 LOAD utility.

Specifying the default

You can specify the default for the DSNPAT command option in your installation options module by using the DSNPAT installation option. LOADPLUS was shipped with a different default value for each DDTYPE. The command option overrides the default that is in the installation options module.

'pattern'

Specify the pattern of variables and text, with surrounding single quotes, to use to create data set names.

The pattern that you specify in your DSNPAT option must allow LOADPLUS to generate unique data set names. If LOADPLUS encounters non-unique data set names, processing terminates. Note the following considerations:

- With multiple SORTOUT and SYSUT1 files, you must include the &DDNAME variable to generate unique names.
- For copy data sets, you might need to include additional variables, such as &VCAT, &DATEJ, or &TIME4 to generate unique names across multiple loads.
- If you plan to run concurrent LOADPLUS jobs or run multiple LOADPLUS steps in a worklist environment, BMC recommends that you include additional
variables in the default pattern to ensure unique names. For example, the following value ensures that LOADPLUS generates unique names:

```
DSNPAT 'UID.UTILPFX..DDNAME..DATE..TIME'
```

**Variables**

You can use any of the symbolic variables that are described in Table 95 on page 415 to construct your pattern. In addition, you can use text or provide user-defined variables from a user exit routine. For more information about these user exit routines, see the LOAD command option, “DSNUEXIT” on page 395, and “LOADPLUS user exits” on page 825.

Symbols for numeric variables (such as &DATE or &TIME) must be prefixed by a national character (alphabetic, #, @, or $). In the following example, the first statement causes an error, while the second is correct:

```
DSNPAT '&DB.&TS..DATE'
DSNPAT '&DB..RP.DATE'
```

Although you can *prefix* a symbolic variable with an alphabetic character, you cannot *append* characters. For example, XX&DB is valid, but &DBXX is invalid. &DB.XX is valid.

User-defined variables must begin with an underscore character, as in _DEPT. User-defined variables are not valid for a DSNUTILB load. For more information, see “Rules for the user-defined variables created in LOADPLUS user exits” on page 827.

**Table 95: Symbolic variables for DSNPAT command option**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
<th>Length of result</th>
<th>DSNUTILB load</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;DATE</td>
<td>Current date (in the format MMDDYY)</td>
<td>6 bytes</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;DATEJ</td>
<td>Current Julian date (in the format YYYYDDD)</td>
<td>7 bytes</td>
<td>Variable translated to the IBM &amp;JDATE variable</td>
</tr>
<tr>
<td>&amp;DB</td>
<td>Database containing the space being used for this data set allocation</td>
<td>8 bytes maximum</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;DDNAME</td>
<td>ddname being used for this data set allocation</td>
<td>8 bytes maximum</td>
<td>Value passed</td>
</tr>
<tr>
<td>&amp;GRPNM</td>
<td>DB2 data sharing group name, or, in a non-data-sharing environment, the DB2 SSID</td>
<td>4 bytes</td>
<td>Value passed</td>
</tr>
<tr>
<td>&amp;JDATE</td>
<td>Current Julian date (in the format YYDDD)</td>
<td>5 bytes</td>
<td>Variable translated to the IBM &amp;JDATE(3,5) variable</td>
</tr>
<tr>
<td>&amp;JOBNAME</td>
<td>JOB name used in the JCL</td>
<td>8 bytes maximum</td>
<td>Variable passed</td>
</tr>
<tr>
<td>Variable</td>
<td>Definition</td>
<td>Length of result</td>
<td>DSNUTILB load</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
<td>-----------------</td>
<td>---------------</td>
</tr>
<tr>
<td>&amp;PART</td>
<td>Partition being used for this copy data set allocation You can use this variable for any data set. However, LOADPLUS substitutes the partition number only for copy data sets. For all other data sets, LOADPLUS substitutes the value 000.</td>
<td>3 bytes for table spaces with 999 partitions or less 4 bytes for table spaces with 1000 through 4096 partitions</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;REPLACE</td>
<td>LOAD REPLACE being done (Y or N)</td>
<td>1 byte</td>
<td>Job terminated</td>
</tr>
<tr>
<td>&amp;RESUME</td>
<td>LOAD RESUME being done (Y or N)</td>
<td>1 byte</td>
<td>Job terminated</td>
</tr>
<tr>
<td>&amp;SSID</td>
<td>DB2 subsystem ID</td>
<td>4 bytes</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;STEPNAME</td>
<td>STEP name used in the JCL LOADPLUS ignores PROC names.</td>
<td>8 bytes maximum</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;TIME</td>
<td>Current time (in the format HHMMSS)</td>
<td>6 bytes</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;TIME4</td>
<td>Current time (in the format HHMM)</td>
<td>4 bytes</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;TS</td>
<td>Table space containing the table specified in your LOAD command</td>
<td>8 bytes maximum</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;USERID or &amp;UID</td>
<td>Job user ID You must have a security package to use this variable.</td>
<td>8 bytes maximum</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;UTIL</td>
<td>BMC utility ID LOADPLUS truncates utility IDs longer than 8 bytes</td>
<td>8 bytes maximum</td>
<td>Variable translated to the IBM &amp;UTILID variable</td>
</tr>
<tr>
<td>&amp;UTILPFX</td>
<td>BMC utility ID prefix</td>
<td>8 bytes maximum</td>
<td>Value passed</td>
</tr>
<tr>
<td>&amp;UTILSFX</td>
<td>BMC utility ID suffix</td>
<td>8 bytes maximum</td>
<td>Value passed</td>
</tr>
<tr>
<td>&amp;VCAT</td>
<td>VCATNAME specified in the DB2 catalog for the table space that you are loading; or, if the table space is partitioned, the VCAT name from the first partition that you are loading</td>
<td>8 bytes</td>
<td>Job terminated</td>
</tr>
</tbody>
</table>

Utility IDs that include special characters might cause LOADPLUS to generate invalid data set names. For more information, see "Utility ID variable" on page 397.
If the utility ID has no special character delimiters, &UTILPFX contains the first eight bytes of text and &UTILSFX contains the remaining eight bytes of text.

If the utility ID has a special character delimiter within the first eight bytes of text, &UTILPFX contains the bytes up to but not including the delimiter. &UTILSFX contains the eight bytes following the first delimiter. The first delimiter is not included in either variable. Any delimiter after the first is treated as normal text and might be included in &UTILSFX. Depending on the utility ID that is specified for this load job, &UTILSFX might be blank.

For example, if the utility ID is USER1/WORK1, the variables contain the following values:

&UTIL = USER1/WO
&UTILPFX = USER1
&UTILSFX = WORK1

If the utility ID is USER1//WORK1, the variables contain the following values:

&UTIL = USER1//W
&UTILPFX = USER1
&UTILSFX = /WORK1

The following special delimiter characters tell LOADPLUS to split the utility ID:

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>Period</td>
</tr>
<tr>
<td>+</td>
<td>Plus sign</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>;</td>
<td>Semicolon</td>
</tr>
<tr>
<td>-</td>
<td>Dash</td>
</tr>
<tr>
<td>/</td>
<td>Slash</td>
</tr>
<tr>
<td>\</td>
<td>Broken bar</td>
</tr>
<tr>
<td>_</td>
<td>Underscore</td>
</tr>
<tr>
<td>:</td>
<td>Colon</td>
</tr>
<tr>
<td>=</td>
<td>Equal sign</td>
</tr>
</tbody>
</table>

**Note**
Any other special characters in the utility ID might cause LOADPLUS to generate invalid data set names.

**Name construction**
You can specify any or all nodes of a data set name by using variables or text. The following example generates data set names that contain the ID of the user, the table that is involved in the load, and the name of the load job:

```
DSNPAT '&UID.&TS.&JOBNAME'
```

The following example combines actual text with symbolic variables to generate a data set name:

```
DSNPAT '&DB.&DDNAME..NEW'
```

In certain cases, a period is required in your pattern as a node delimiter or to indicate the end of a variable name. The following table illustrates concatenation in a data set name pattern and those instances in which a period is needed.

**Note**
Concatenation is not affected by a period contained in the value of a user variable.

**Table 97: Variable concatenation examples**

<table>
<thead>
<tr>
<th>Task</th>
<th>Code</th>
<th>Result (where &amp;UID=RDAB and _DEPT=DEV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concatenate the values of 2 variables, no node delimiter.</td>
<td>&amp;UID_DEPT</td>
<td>RDABDEV</td>
</tr>
<tr>
<td>Make 2 nodes from the values of 2 variables.</td>
<td>&amp;UID._DEPT</td>
<td>RDAB.DEV</td>
</tr>
<tr>
<td>Concatenate the value of a variable with text, no node delimiter.</td>
<td>&amp;UID.NEW</td>
<td>RDABNEW</td>
</tr>
<tr>
<td>Concatenate text with the value of a variable, no node delimiter.</td>
<td>NEW&amp;UID</td>
<td>NEW.RDAB</td>
</tr>
<tr>
<td>Make 2 nodes from the value of a variable followed by text</td>
<td>&amp;UID..NEW</td>
<td>RDAB.NEW</td>
</tr>
</tbody>
</table>

LOADPLUS ignores trailing blanks and null value variables. Node-delimiting periods in the pattern, however, are included regardless of the variable’s value. This can result in an invalid data set name.

For example, given that &UID=RDAB and the value of _DEPT is null, the following pattern results in an invalid data set name of RDAB..NEW:

```
&UID._DEPT..NEW
```
**GDG names**

For the following dynamically allocated data sets, you can also specify a pattern that contains a GDG name:

- Copy data sets
- SYSDISC files
- SYSERR files
- SYSMAP files

Each DDTYPE must have a different GDG base.

The GDG format that you use to construct data set names is the same as the one that you use in your JCL when allocating data sets with DD statements. Append the generation number in parentheses. The open parenthesis tells LOADPLUS that the pattern is a GDG name. The generation number can be an integer from 1 through 255.

If the base does not exist, LOADPLUS creates it for you, using everything in the pattern up to the open parenthesis as the base name. For more information, see “Generating data set names in LOADPLUS” on page 80.

The following example shows a valid GDG name:

```&UTILPFX.&DDNAME..COPY(+1)```

If you use a substitution variable as the last variable before the open parenthesis, include a period before the open parenthesis, as in the following example:

```&UTILPFX.&DDNAME.(+1)```

For copy data sets, each partition must have a different GDG base if you specify COPYLVL PART on the LOAD command. To specify a pattern that includes a partition, the partition must not be in parentheses. The following example shows a valid name:

```&UTILPFX.&DDNAME..P&PART.(+1)```

You cannot specify a pattern that contains a PDS name. The following example shows an invalid name:

```&UTILPFX.&DDNAME..(P&PART)```

`'NONE' or ''`

This value indicates that you do not want to use a pattern to create data set names for dynamic allocation. This option is valid only for DDTYPE SORTWORK.
WARNING

If you specify NONE for SORTOUT, SYSUT1, SYSDISC, SYSERR, or copy data set patterns, LOADPLUS does not allocate your data sets and processing terminates.

You must enclose the keyword NONE with single quotes.

Note

In the DSNPAT installation option, the keyword NONE must not be enclosed by single quotes.

EXPDT

Specify EXPDT to set an expiration date for certain data sets that you are dynamically allocating.

```
EXPDT date
```

This option applies to the following data sets:

- Copy data sets
- SYSDISC data sets
- SYSERR data sets
- SYSMAP data sets

LOADPLUS ignores this option for any other data sets.

The value of `date` must be in the format `yyyyddd` or `yyyy/ddd`. The variable `yyyy` is the 4-digit year (1900 through 2155) and `ddd` is the 3-digit Julian day (000 through 366).

Additional considerations

The following considerations apply to the EXPDT option:

- If DELETEFILES YES is in effect, LOADPLUS deletes your SYSERR data set only if the expiration date is earlier than the current date.

- If DELETEFILES YES SYSDISC YES is in effect and you have no discarded records, LOADPLUS deletes your SYSDISC data set only if the expiration date is earlier than the current date.

- When invoking DSNUTILB, LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. For
more information about TEMPLATE control statements, see the documentation for the DB2 LOAD utility.

**Specifying the default**

You can specify the default for the EXPDT command option in your installation options module by using the EXPDT installation option. LOADPLUS was shipped without a value for this option. The EXPDT command option overrides the RETPD command option and the EXPDT and RETPD installation options.

**RETPD**

Specify RETPD to set a retention period (in days) for certain data sets that you are dynamically allocating.

```
RETPD integer
```

This option applies to the following data sets:

- Copy data sets
- SYSDISC data sets
- SYSERR data sets
- SYSMAP data sets

LOADPLUS ignores this option for any other data sets.

The value of `integer` must be in the range 0 through 9999.

**Additional considerations**

The following considerations apply to the RETPD option:

- If DELETEFILES YES is in effect, LOADPLUS deletes your SYSERR data sets only if RETPD is 0 or blank.

- If DELETEFILES YES SYSDISC YES is in effect and you have no discarded records, LOADPLUS deletes your SYSDISC data set only if RETPD is 0 or blank.

- When invoking DSNUTILB, LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. For more information about TEMPLATE control statements, see the documentation for the DB2 LOAD utility.

**Specifying the default**
You can specify the default for the RETPD command option in your installation options module by using the RETPD installation option. LOADPLUS was shipped without a value for this option. The RETPD command option overrides the default that is in the installation options module for both the RETPD and EXPDT installation options. However, if you specify the EXPDT command option, it takes precedence over the RETPD command option.

**GDGLIMIT**

Specify GDGLIMIT to indicate the number of generations to keep for certain data sets that you are dynamically allocating as a GDG data set.

```
GDGLIMIT 5
```

This option applies to the following data sets:

- Copy data sets
- SYSDISC data sets
- SYSERR data sets
- SYSMAP data sets

The value of `integer` must be in the range 1 through 255.

LOADPLUS honors this option only when creating the GDG base. LOADPLUS ignores this option for any data sets other than the ones listed.

When invoking DSNUTILB, LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. For more information about TEMPLATE control statements, see the documentation for the DB2 LOAD utility.

**Specifying the default**

You can specify the default for the GDGLIMIT command option in your installation options module by using the GDGLIMIT installation option. LOADPLUS was shipped with a default value of 5 for this option. The command option overrides the default that is in the installation options module.

**SMS class block**

These options allow you to specify the classes to use for SMS allocations.
You can repeat this block to specify more than one type of SMS class.

You must enclose the class names for each of the class types in parentheses and you must separate them with a comma. Use the keyword NONE instead of the class name if you do not want to specify a class for dynamic data set allocation.

The THRESHLD installation option or command option determines how the utility selects the classes that you specify here. This value applies to the secondary device type that you specify with the UNIT option (or, if applicable, through SMS).

**DSNUTILB reorganization jobs**

When invoking DSNUTILB, the utility includes these classes in the TEMPLATE control statement that it builds for the IBM utility.

If building a secondary TEMPLATE control statement for your copy data sets, the utility includes the secondary classes in that template. For information about building a secondary template, see the THRESHLD option.

**Specifying the default**

You can specify the default for the SMS class block in your installation options module by using the DATACLAS, MGMTCLAS, or STORCLAS installation options. The utility was shipped with default values of (NONE,NONE) for these options. The command options override the default that is in the installation options module.

**DATACLAS**

DATACLAS indicates the primary and secondary SMS data classes to use for allocating the specified data set type. For class1 and class2, specify valid SMS data class names that do not exceed eight characters each.
MGMTCLAS

MGMTCLAS indicates the primary and secondary SMS management classes to use for allocating the specified data set type. For *class1* and *class2*, specify valid SMS management class names that do not exceed eight characters each.

STORCLAS

STORCLAS indicates the primary and secondary SMS storage classes to use for allocating the specified data set type. For *class1* and *class2*, specify valid SMS storage class names that do not exceed eight characters each.
Building and executing LOADPLUS jobs

This chapter describes the elements of a LOADPLUS job, including the data sets that LOADPLUS uses. This chapter also tells you how to run jobs and what to do if a job terminates or is canceled.

Building the LOADPLUS job

Building a job for the LOADPLUS for DB2 product involves creating a set of JCL that includes the following elements:

■ A JOB statement ("JOB statement" on page 425)
■ An EXEC statement with the appropriate utility parameters ("LOADPLUS EXEC statement" on page 426)
■ STEPLIB or JOBLIB DD statements ("LOADPLUS STEPLIB DD statement" on page 431)
■ DD statements for the appropriate number and size of data sets ("LOADPLUS DD statements" on page 432)
■ LOADPLUS control statements using the appropriate command syntax ("Syntax of the LOAD command" on page 163)

For examples of LOADPLUS jobs, see "Examples of LOADPLUS jobs" on page 483.

JOB statement

Include a JOB statement that conforms to your site’s standards.

You can include the REGION parameter on either your JOB statement or your EXEC statement. For recommendations, see "REGION parameter" on page 426.
**LOADPLUS EXEC statement**

The LOADPLUS EXEC statement specifies the module to be executed for the LOADPLUS utility.

The LOADPLUS module name is AMUUMAIN. The EXEC statement also specifies LOADPLUS utility parameters, which are described in “Utility parameters on the LOADPLUS EXEC statement” on page 427.

You can use the REGION parameter to specify the region size on either your EXEC statement or your JOB statement. See “REGION parameter” on page 426 for recommendations.

When you are loading a large number of partitions and dynamic allocation is active, BMC recommends that you specify DYNAMNBR=1600 on the EXEC statement of your execution JCL. Including this parameter enables the system to acquire adequate resources immediately before their use and to release them immediately after use.

**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 the optimal amount available virtual storage to the utility job. If your data center does not permit you to specify REGION=0M, specify the amount that allows the most virtual storage both above and below the 16-megabyte line. Using a region size that is less than optimal risks the following potential issues:

- Running less efficiently, which could result in additional CPU and elapsed time
- Encountering memory failures or jobs that fail when new versions implement changes that require additional memory

**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.

- Specify NOLIMIT to allow unlimited above-the-bar memory.
- If you are unable to specify NOLIMIT, specify at least 4 GB; if you are operating on LOB or XML data, specify at least 32 GB.
Utility parameters on the LOADPLUS EXEC statement

The LOADPLUS EXEC statement includes the following utility parameters:

- DB2 subsystem ID or group attachment name
- Utility ID
- Restart parameter
- TSO user ID for notification of progress made on the load
- Message level
- Installation options module

The following illustration shows the format of the EXEC statement:

```plaintext
//stepName EXEC PGM=AMUUMAIN,
//PARM='ssid,utilityID,restartParameter,userID,MSGLEVEL(n),optionsModule'
```

The LOADPLUS utility parameters are positional. If you do not specify a value for a parameter (thus allowing the default value), you must substitute a comma for that parameter if additional parameters follow. The comma indicates that a parameter was omitted.

**DB2 subsystem identifier (SSID)**

This parameter specifies the four-character DB2 subsystem ID (SSID) that indicates where the table space resides.

If you do not specify the SSID, LOADPLUS uses the DB2 installation default from the DSNHDECP module. LOADPLUS depends on the application defaults module being named DSNHDECP. If you do not specify an SSID and LOADPLUS cannot find a module named DSNHDECP in your LINKLIST or STEPLIB, LOADPLUS terminates.

The following considerations apply when running LOADPLUS in a data sharing environment:

- When you supply a group attachment name as the SSID, LOADPLUS uses the name to connect all plans. LOADPLUS then determines the actual DB2 SSID from within that group to use for the current load job.
- When restarting in a data sharing environment, LOADPLUS might use any member of the specified group.
- When both of the following conditions exist, specify a member SSID for your load job instead of a group attachment name. Ensure that the DB2 version of the SSID that you specify corresponds to the version of the DB2 load library that you specify in your STEPLIB.
  
  — You are using table space compression.

---

Chapter 4  Building and executing LOADPLUS jobs 427
The subsystems in your data sharing group are not all at the same DB2 version.

**Utility identifier (utility ID)**

This parameter specifies the 1- to 16-character utility ID that gives a unique name to a utility job. If you omit this parameter, LOADPLUS uses the default, *userId.jobName*. Each BMC Software utility job should have a unique utility ID.

**Note**

Utility IDs that include special characters might cause LOADPLUS to generate invalid data set names when using dynamic allocation. For more information, see the discussion about using the utility ID variable with the DSNPAT option on “DSNPAT” on page 414.

**Restart parameter**

The restart parameter can have one of the following values:

- Blank or not specified
- RESTART
- RESTART(PHASE)
- NEW
- NEW/RESTART
- NEW/RESTART(PHASE)
- TERM
- MAINT

For more information about specific circumstances for which you should use a particular value, see “Restarting LOADPLUS” on page 467.

For DSNUTILB jobs, LOADPLUS passes the restart parameter to DSNUTILB after the UTILINIT phase. Restart processing is handled by the IBM DB2 LOAD utility.

**Blank or not specified**

By not specifying a restart parameter, LOADPLUS initiates a new BMC utility job. The utility ID that you specify cannot currently exist in the BMCUTIL table.

**RESTART**

With the following exception, specifying this value restarts a BMC utility from the last restart sync point. LOADPLUS takes restart sync points as each phase completes and as the processing of each DB2 object completes.
The utility ID must exist in the BMCUTIL table. If the utility ID does not exist in the BMCUTIL table, LOADPLUS ends with return code 8.

**Exception:** For partition-by-growth table spaces when the LOAD or COMBINED phase has not completed, LOADPLUS restarts at the first new row written during the original job. LOADPLUS starts the copy process with the first copy that did not complete.

**RESTART(PHASE)**

For a two-phase load, specifying this value restarts LOADPLUS at the beginning of the last incomplete phase. In general, for a single-phase load or for an SQLAPPLY load, LOADPLUS treats RESTART(PHASE) as if you specified RESTART.

The utility ID must exist in the BMCUTIL table. If the utility ID does not exist in the BMCUTIL table, LOADPLUS ends with return code 8.

**Exception:** For partition-by-growth table spaces when the LOAD phase has not completed, LOADPLUS restarts at the first new row written during the original job. LOADPLUS starts the copy process with the first copy that did not complete.

**NEW**

Specifying this value initiates a new BMC utility job or replaces an existing utility ID. Specifying this value allows you to start a utility without having to end the utility ID separately.

**WARNING**

Use care when you specify NEW. When you replace or terminate a utility ID for LOADPLUS, data from the PRELOAD or COMBINED phase is lost and existing data in the table space can also be lost.

If you specify NEW and the utility ID has a status of X (executing), LOADPLUS issues error message BMC50012E and ends with return code 8.

**NEW/RESTART**

With the following exception, specifying this value restarts a BMC utility from the last restart sync point if the utility ID exists. Otherwise, specifying this value starts a utility as NEW. LOADPLUS takes restart sync points as each phase completes and as the processing of each DB2 object completes.

**Exception:** For partition-by-growth table spaces when the LOAD or COMBINED phase has not completed, LOADPLUS restarts at the first new row written during the original job. LOADPLUS starts the copy process with the first copy that did not complete.
NEW/RESTART(PHASE)

For a two-phase load, specifying this value restarts LOADPLUS at the beginning of the last incomplete phase if the utility ID exists. Otherwise, for a two-phase load, specifying this value starts LOADPLUS as NEW.

**Exception:** For partition-by-growth table spaces when the LOAD phase has not completed, LOADPLUS restarts at the first new row written during the original job. LOADPLUS starts the copy process with the first copy that did not complete.

In general, for a single-phase load or for an SQLAPPLY load, LOADPLUS treats NEW/RESTART(PHASE) as if you specified NEW/RESTART.

TERM

Specifying this value terminates the existing utility, removes the utility ID from the BMCUTIL table, and removes the corresponding rows from the BMCSYNC and BMCDICT tables. After removing all sync point and restart information, LOADPLUS terminates without performing a load. LOADPLUS terminates with return code 0, regardless of whether the utility ID exists.

For a DSNUTILB load, LOADPLUS also terminates an existing DB2 utility ID.

When specifying TERM, you need only minimal JCL. Your JCL must include at least a SYSPRINT DD statement and STEPLIB to the LOADPLUS load library.

**WARNING**

Use care when you specify TERM. When you replace or terminate a utility ID, data from the PRELOAD phase or the COMBINED phase is lost and existing data in the table space can also be lost.

MAINT

Specifying this value forces MSGLEVEL(1) and causes LOADPLUS to print the following information:

- An options module report that lists the values in the installation options module that you are using
- The values in the DSNHDECP module that LOADPLUS uses (which is the DSNHDECP module that DB2 loads)
- A summary report of all of the product fixes that you have applied

When you specify the MAINT parameter, the job ends without affecting any utility that is running.
For this parameter, you need only minimal JCL. Your JCL must include at least a SYSPRINT DD statement and STEPLIB to the LOADPLUS and DB2 load libraries.

**User identifier (user ID)**

This parameter specifies the TSO user ID that LOADPLUS notifies after the completion of each phase and at the end of utility command execution.

**Message level (MSGLEVEL)**

This parameter controls which messages LOADPLUS returns in the SYSPRINT and SYSPRIN2 data sets. MSGLEVEL(0) returns minimal messages. MSGLEVEL(1) returns additional messages to help you diagnose problems and fine-tune performance.

You can use the MSGLEVEL installation option to change the default value of this parameter. For details, see “MSGLEVEL=1” on page 741.

**Installation options module**

This parameter allows you to identify which installation options module to use. If you include this parameter, you must specify the full name of the options module. If you omit this parameter, LOADPLUS uses the default installation options module, AMU$OPTS.

For more information about installation options, see “Overview of LOADPLUS installation options” on page 703. For information about how to create multiple installation options modules, see the Installation System documentation.

**LOADPLUS STEPLIB DD statement**

The LOADPLUS STEPLIB DD statement must specify the following libraries, unless they are included in your system’s LINKLIST or in a JOBLIB statement:

- Load libraries that contain the files (including the options modules) for the following BMC products and components:
  - LOADPLUS
  - BMCSORT (AUP)
  - Common Statistics (ATS)
  - DB2 Utilities Common Code (D2U)
  - DB2 Solution Common Code (SCC)
  - High-speed Apply, if you are performing an SQLAPPLY load
- SAS/C transient library, if you specify BMCSTATS YES, BMCSTATS REPORT, or UPDATEDB2STATS YES
- BMCPSWD and ASUBMAIN libraries, if you are updating statistics in the DASD MANAGER PLUS statistics tables
- Libraries that contain any DB2 user exits (EDITPROCs, VALIDPROCs, FIELDPROCs, and user-written conversion routines)
- DB2 load library

All load libraries in the STEPLIB or JOBLIB concatenation must be APF authorized.

LOADPLUS DD statements

This topic provides specification guidelines, allocation information, and usage notes for each of the data sets that LOADPLUS uses.

LOADPLUS uses data sets that are specified by ddnames. Use the following table to quickly find the data set for which you want more information. This table also provides a quick reference to associated command and installation options for each data set type.

Table 98: Data set type descriptions and quick command reference

<table>
<thead>
<tr>
<th>Data set type</th>
<th>Reference</th>
<th>Default ddname or prefix</th>
<th>DDTYPE option keyword (<em>DDTYPE</em> on page 397)</th>
<th>Ddbname or prefix installation option</th>
<th>Ddbname or prefix command option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command input</td>
<td>“SYSIN data sets” on page 454</td>
<td>SYSIN</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Data set type</td>
<td>Reference</td>
<td>Default ddname or prefix</td>
<td>DDTYPE option keyword (&quot;DDTYPE&quot; on page 397)</td>
<td>ddname or prefix installation option</td>
<td>ddname or prefix command option</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
<td>--------------------------</td>
<td>-----------------------------------------------</td>
<td>--------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Copy</td>
<td>&quot;Copy data sets in LOADPLUS&quot; on page 436</td>
<td>■ BMCCPY (local primary) ■ BMCCPZ (local backup) ■ BMCRCY (remote primary) ■ BMCRCZ (remote backup) ■ LLP (local primary for LOB) ■ LLB (local backup for LOB) ■ LRP (remote primary for LOB) ■ LRB (remote backup for LOB) ■ XLP (local primary for XML) ■ XLB (local backup for XML) ■ XRP (remote primary for XML) ■ XRB (remote backup for XML)</td>
<td>■ LOCPFCPY ■ LOCBFCPY ■ REMPFCPY ■ REMBFCPY ■ LOCPLCPY ■ LOCBLCPY ■ REPLCPY ■ REMBLCPY ■ LOCXPACY ■ LOCBXCPY ■ REMPXCPY ■ REMBXCOPY</td>
<td>COPYDDN (primary copies) (&quot;COPYDDN=(BMCCPY, BMCCPZ)&quot; on page 718) RCVYDDN (remote copies) (&quot;RCVYDDN=(BMCRCY, BMCRCZ)&quot; on page 744) These options are not available for LOB or XML copy data sets.</td>
<td>COPYDDN (primary copies) (&quot;COPYDDN&quot; on page 300) RECOVERYDDN (remote copies) (&quot;RECOVERYDDN&quot; on page 302) These options are not available for LOB or XML copy data sets.</td>
</tr>
<tr>
<td>Discard</td>
<td>&quot;SYSDISC data set in LOADPLUS&quot; on page 450</td>
<td>SYSDISC</td>
<td>DISCARD</td>
<td>DISCDDN a (&quot;DISCDDN=SYSDISC&quot; on page 723)</td>
<td>DISCARDDDN a (&quot;DISCARDDN&quot; on page 217)</td>
</tr>
<tr>
<td>Data set type</td>
<td>Reference</td>
<td>Default ddname or prefix</td>
<td>DDTYPE option keyword (&quot;DDTYPE&quot; on page 397)</td>
<td>ddname or prefix installation option</td>
<td>ddname or prefix command option</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------------------------</td>
<td>--------------------------</td>
<td>-----------------------------------------------</td>
<td>---------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Error</td>
<td>“SYSERR data set in LOADPLUS” on page 453</td>
<td>SYSERR</td>
<td>ERROR</td>
<td>ERRDDN&lt;sup&gt;a&lt;/sup&gt;</td>
<td>ERRDDN&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(&quot;ERRDDN=SYSERR&quot; on page 727)</td>
<td>(&quot;ERRDDN&quot; on page 216)</td>
</tr>
<tr>
<td>IDCAMS input</td>
<td>“SYSIDCIN data set in LOADPLUS” on page 462</td>
<td>SYSIDCIN</td>
<td>Not applicable</td>
<td>IDCDDN&lt;sup&gt;a&lt;/sup&gt;</td>
<td>IDCDDN&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(&quot;IDCDDN=SYSIDCIN&quot; on page 732)</td>
<td>(&quot;IDCDDN&quot; on page 249)</td>
</tr>
<tr>
<td>Index work</td>
<td>“SYSUT1 data sets in LOADPLUS” on page 459</td>
<td>SYSUT1</td>
<td>WORK</td>
<td>WORKDDN&lt;sup&gt;a&lt;/sup&gt;</td>
<td>WORKDDN&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(&quot;WORKDDN=SYSUT1&quot; on page 757)</td>
<td>(&quot;WORKDDN&quot; on page 215)</td>
</tr>
<tr>
<td>Input</td>
<td>“SYSREC data sets in LOADPLUS” on page 456</td>
<td>SYSREC</td>
<td>Not applicable</td>
<td>INDDN&lt;sup&gt;a&lt;/sup&gt;</td>
<td>INDDN&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(&quot;INDDN=SYSREC&quot; on page 757)</td>
<td>(&quot;INDDN&quot; on page 212)</td>
</tr>
<tr>
<td>Load work</td>
<td>“SORTOUT data sets in LOADPLUS” on page 443</td>
<td>SORTOUT</td>
<td>LOAD</td>
<td>LOADDN&lt;sup&gt;a&lt;/sup&gt;</td>
<td>LOADDN&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(&quot;LOADDN=SORTOUT&quot; on page 736)</td>
<td>(&quot;LOADDN&quot; on page 213)</td>
</tr>
<tr>
<td>Mapping file</td>
<td>“SYSMAP data set in LOADPLUS” on page 454</td>
<td>SYSMAP</td>
<td>SYSMAP</td>
<td>MAPDDN&lt;sup&gt;a&lt;/sup&gt;</td>
<td>MAPDDN&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(&quot;MAPDDN=SYSMAP&quot; on page 738)</td>
<td>(&quot;MAPDDN&quot; on page 284)</td>
</tr>
<tr>
<td>Message output</td>
<td>“SYSPRINT data sets” on page 455</td>
<td>SYSPRINT</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Message output</td>
<td>“SYSPRIN2 data sets” on page 455</td>
<td>SYSPRIN2</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Other</td>
<td>“Other data sets” on page 465</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>SNAP dump output for SQLAPPLY load</td>
<td>“APTDUMP data sets in LOADPLUS” on page 435</td>
<td>APTDUMP</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>
For an alternative to specifying data sets in DD statements in your JCL, you can have LOADPLUS dynamically allocate the following types of data sets:

- SYSREC
- Copy data sets
- SORTOUT
- SORTWK
- SYSDISC
- SYSERR
- SYSMAP
- SYSUT1

For details about dynamically allocating SYSREC data sets, see “INDSN” on page 211. For details about dynamically allocating the other data sets, see “Dynamic data set allocation in LOADPLUS” on page 78 and “Dynamic allocation options” on page 393.

**APTDUMP data sets in LOADPLUS**

The APTDUMP data set contains output from a SNAP dump if the High-speed Apply Engine generates a dump in response to internal errors during an SQLAPPLY load.

BMC recommends including this optional data set in your JCL as a diagnostic aid.
ASUSRPT data sets

You can specify an ASUSRPT DD statement in your JCL to allocate a data set to contain the statistics reporting output from the Common Statistics component.

Note
The Common Statistics component writes informational and diagnostic messages to SYSPRINT, regardless of whether you specify this DD statement in your JCL.

Alternatively, you can use one of the following options for your statistics output:

- Send it to SYSPRINT.
  To use this method, do not specify an ASUSRPT DD statement in your JCL. The utility automatically sends the report to your SYSPRINT.
  
  Note
  The Common Statistics component does not send this output to a SYSPRIN2 data set.

- Suppress the output.
  To suppress the output, specify the following DD statement in your JCL:
  
  //ASUSRPT DD DUMMY

BMCFORCE data sets

You can specify a BMCFORCE DD statement in your JCL to allocate a data set to contain the thread cancelation report output that is generated when you specify one of the following options:

- FORCE REPORTONLY
- FORCE_RPT YES with FORCE READERS or FORCE ALL

If you do not specify a BMCFORCE DD statement in your JCL, the utility sends the report to your SYSPRINT.

Copy data sets in LOADPLUS

Copy data sets are output data sets that contain a DSN1COPY-type copy or image copy.

LOADPLUS creates one or more output copy data sets when you specify COPY YES. Depending on the option that you choose, LOADPLUS creates either a standard image copy or DSN1COPY-type copy after loading the data into the table, or an inline image copy while loading the data into the table. LOADPLUS creates all copies with system pages at the beginning of the data set (in the same way that the IBM COPY utility creates image copies when you specify SYSTEMPAGES YES).
The copy ddnames identify the output data sets that will contain the copy either of the table space, of each partition in the table space, or of the specified subset of partitions after the load. The number of copies that LOADPLUS makes when you specify COPY YES depends on the following information:

- (without dynamic allocation) The existence of the copy ddnames in your JCL
- (with dynamic allocation) The DDTYPE and COPYLVL specifications in your installation and command options

LOADPLUS provides the following default names for each type of copy data set. With the exception of the LOB and XML copy data sets, you can use a different name by changing it in your installation or command options.

Table 99: Default copy data set names

<table>
<thead>
<tr>
<th>Default name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMCCPYnn</td>
<td>Local primary</td>
</tr>
<tr>
<td>BMCCPZnn</td>
<td>Local backup</td>
</tr>
<tr>
<td>BMCRCYnn</td>
<td>Remote primary</td>
</tr>
<tr>
<td>BMCRCZnn</td>
<td>Remote backup</td>
</tr>
<tr>
<td>LLPnnnnn</td>
<td>Local primary LOB</td>
</tr>
<tr>
<td>LLBnnnnn</td>
<td>Local backup LOB</td>
</tr>
<tr>
<td>LRPnnnnn</td>
<td>Remote primary LOB</td>
</tr>
<tr>
<td>LRBnnnnn</td>
<td>Remote backup LOB</td>
</tr>
<tr>
<td>XLPnnnnn</td>
<td>Local primary XML</td>
</tr>
<tr>
<td>XLBnnnnn</td>
<td>Local backup XML</td>
</tr>
<tr>
<td>XRPnnnnn</td>
<td>Remote primary XML</td>
</tr>
<tr>
<td>XRBnnnnn</td>
<td>Remote backup XML</td>
</tr>
</tbody>
</table>

Requirements

LOADPLUS requires the following copy data sets when you specify COPY YES:

- When you are loading LOB data, BMCCPY is required for the base table copy and LLP is required for the LOB table space copy.
- When you are loading XML data, BMCCPY is required for the base table copy and XLP is required for the XML table space copy.
- For all other load jobs, BMCCPY is required.
- For all load jobs, if you specify BMCRCZ, BMCRCY is also required.
LOADPLUS requires that your copy data sets be dynamically allocated in either of the following cases:

- You are running a DSNUTILB load.
- You are making LOB or XML table space copies. (Data sets for base table copies do not need to be dynamically allocated.)

**Restrictions**

The LOB and XML copy data set prefixes (see Table 99 on page 437) are reserved. If you include a DD statement in your JCL that uses these prefixes as the data set name or prefix, LOADPLUS terminates.

**Overriding the default ddnames**

To override the default copy ddnames or ddname prefixes in your installation options module, see the information in Table 98 on page 432.

**Registration**

The ddname itself controls the registration information that is placed in the ICBACKUP column of SYSIBM.SYSCOPY when you specify either ALL (the default) or specific ddnames for the REGISTER command option.

**Methods for allocating copy data sets**

You can use one of the following methods to allocate your copy data sets and determine the appropriate size for those data sets:

- Have LOADPLUS dynamically allocate your copy data sets by using the DDTYPE command or installation option. Note the following considerations about using this method:
  - This method is required for copies for a DSNUTILB load.
  - This method is required for LOB and XML table space copies.
  - BMC recommends this method when you are loading a partition-by-growth table space.

- Specify ANALYZE PAUSE, ANALYZE ONLY, or PRELOAD PAUSE to have LOADPLUS provide an estimate of the space that is needed for the copy data sets. For more information about the options that provide space estimates for you, see “ANALYZE option for estimating data set allocation in LOADPLUS” on page 465 or “ANALYZE” on page 258.

- Calculate the space allocation based on your knowledge of the table space that you are loading and the information in this section.
LOADPLUS determines the optimal block size of the copy data sets based on the device type that contains the data set. Copy data sets can be on different device types. However, if additional copy data sets (for example, remote backup copy data sets) for the same object are on different device types, the block size for all copy data sets for that object is the block size that LOADPLUS determines is optimal for the local primary copy.

**Dynamically allocating copy data sets**

Based on the information in Table 100 on page 440, specify dynamic allocation options that result in the same number and type of copy data sets as if you specified DD statements in your JCL.

**Restriction**

LOADPLUS does not allow dynamically allocated copy data sets to be stacked on tape.

**Additional considerations**

The following considerations apply when dynamically allocating copy data sets:

- For copy data sets, LOADPLUS determines the size of the dynamically allocated files based on the value that you specify for the AUTOENUMROWS installation option or the ENUMROWS command option. Additionally, for LOAD RESUME YES, LOADPLUS takes into account the high-used relative byte address (HURBA) of the table space that you are loading.

- For LOB and XML copy data sets, LOADPLUS does not report allocation information. LOADPLUS allocates these data sets based on LOB or XML record length, number of records, and the following options. To reduce the risk of out-of-space abends, ensure that these options are set appropriately:
  - LOBAVGPCCT
  - XMLAVGSIZE
  - AVGSIZE
  - SIZEPCT

- For information about partition-level copies, see “Partition-level copies” on page 442 and “Partition-by-growth table spaces” on page 443.

**Allocating copy data sets in your JCL**

If you do not use dynamic allocation, you must allocate the copy data sets in your JCL by specifying DD statements as described in the following table:
### Table 100: DD statements required when allocating copy data sets in your JCL

<table>
<thead>
<tr>
<th>Table space being copied</th>
<th>Type of load</th>
<th>Copy DD statements required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nonpartitioned</td>
<td>Any type</td>
<td>■ Specify one DD statement for each copy type that you want LOADPLUS to make.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Do not use \textit{nn} in the ddname.</td>
</tr>
<tr>
<td>Partitioned (except partition-by-</td>
<td>All partitions (entire table space)</td>
<td>Specify one DD statement for each copy type that you want LOADPLUS to make. Do not use \textit{nn} in the ddname.</td>
</tr>
<tr>
<td>growth)</td>
<td></td>
<td>or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For each partition, specify one DD statement for each copy type that you want LOADPLUS to make. Use \textit{nn} for all ddnames, where \textit{nn} matches the partition number.</td>
</tr>
<tr>
<td>Selected partitions (using INTO</td>
<td>If you specify a single subset of</td>
<td>If you specify a single subset of contiguous partitions, you can specify one DD statement for each copy type that you want LOADPLUS to make. Do not use \textit{nn} in the ddname.</td>
</tr>
<tr>
<td>TABLE PART option)</td>
<td>contiguous partitions, you can</td>
<td>or</td>
</tr>
<tr>
<td></td>
<td>specify one DD statement for each</td>
<td>For each specified partition, specify one DD statement for each copy type that you want LOADPLUS to make. Use \textit{nn} for all ddnames, where \textit{nn} matches the partition number. (The \textit{nn} is not required if you are performing a partial load with only one partition.)</td>
</tr>
<tr>
<td></td>
<td>copy type that you want LOADPLUS to make.</td>
<td></td>
</tr>
</tbody>
</table>
Table space being copied | Type of load | Copy DD statements required |
--- | --- | --- |
Partition-by-growth | All partitions (entire table space) | Specify one DD statement for each copy type that you want LOADPLUS to make. Do not use "nn" in the ddname. or For each partition, specify one DD statement for each copy type that you want LOADPLUS to make. Use "nn" for all ddnames, where "nn" matches the partition number. If necessary, specify additional DD statements. |
Selected partitions (using INTO TABLE PART option) | This option is not valid for partition-by-growth table spaces. |

a This option is not valid for a DSNUTILB load.
b The value of the COPYSUBSET installation option must be YES.

The size that LOADPLUS needs for the copy data sets depends on the number of pages that are required when the table space is loaded. For a two-phase load, LOADPLUS calculates the number of required pages and issues message BMC51486I, which includes the number of pages. Use the following table to determine the value to use for your data set allocation. Allocate the primary amount as the total amount. No secondary amount is needed.

**Table 101: Copy data set allocation calculations**

<table>
<thead>
<tr>
<th>Table space type</th>
<th>Number of copy data sets</th>
<th>Calculation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nonpartitioned</td>
<td>Not applicable</td>
<td>Multiply the number of pages required for loading by the page size of the table space.</td>
</tr>
<tr>
<td>Partitioned</td>
<td>Single</td>
<td>Multiply the total number of pages for all loaded partitions by the page size of the table space.</td>
</tr>
<tr>
<td></td>
<td>Multiple</td>
<td>For each data set, multiply the number of pages for that partition by the page size of the table space.</td>
</tr>
</tbody>
</table>

Note the following considerations when allocating copy data sets in your JCL:

- If you specify VOL=(,RETAIN) in your JCL, LOADPLUS does not deallocate the copy data set to free the device after completing the copy.

- You should not specify DISP=MOD for an image copy data set because LOADPLUS does not reset the data set to empty but appends to any data already present in the file.
• Copy data sets should not be temporary data sets. For information about how LOADPLUS defines a temporary data set, see “Work file validity and integrity checks in LOADPLUS” on page 85.

**Partition-level copies**

Note the following additional information when allocating partition-level copies.

*Naming partition-level copies*

Use the *nn* as part of the ddname only if you are loading a partitioned table space and you want a separate data set for each partition.

JCL rules limit ddnames to eight characters. If you are making partition-level copies, the ddname has two parts: the ddname prefix (minimum of one character) and the partition number *nn* (one to seven characters). The ddname that you specify in the installation options or the command options becomes the ddname prefix. Note the following information about the partition number:

• If you are dynamically allocating copy data sets (and you specify COPYLVL PART to make partition-level copies), LOADPLUS appends the partition number to the ddname prefix. If you are loading a table space that contains more than 99 partitions, use the COPYDDN and RECOVERYDDN options to specify a prefix that results in eight characters or less after LOADPLUS appends the highest partition number. (These options do not apply to LOB or XML copy data sets.) For more information, see “Specifying ddname prefixes” on page 81.

• If you are allocating copy data sets in your JCL, you must append the partition number *nn* to the ddname prefix. The *nn* must match the partition that you are copying, and you must allow sufficient bytes for LOADPLUS to add the partition number to the prefix and still have a valid ddname (eight characters or less).

**Large number of partitions**

If you are loading a large number of partitions, consider one of the following options to avoid encountering data set allocation restrictions of the operating system or LOADPLUS memory restrictions:

• Limit the number of copies per partition.

• If you are loading all partitions, dynamically allocate your copy data sets and create a single copy by specifying COPYLVL FULL.

• If you are loading a subset of contiguous partitions, create a single copy by specifying COPYSUBSET=YES in the installation options module and, if you are dynamically allocating your copy data sets, COPYLVL FULL on your LOAD command.
Partition-by-growth table spaces

When you are making partition-level copies for partition-by-growth table spaces, LOADPLUS requires the allocation of enough copy data sets to provide for any additional partitions that it adds during the load. BMC recommends that you dynamically allocate your copy data sets in this case.

When you dynamically allocate your copy data sets, LOADPLUS allocates additional copy data sets only as it needs them. Ensure that the ddname prefix that you specify allows for the number of partitions specified for MAXPARTITIONS in your table space definition.

If you allocate the copy data sets yourself, you must allocate enough data sets to allow for expansion up to the number of partitions specified for MAXPARTITIONS in your table space definition. Although LOADPLUS requires this allocation, it only uses these data sets if it adds partitions to the table space during the load.

SORTOUT data sets in LOADPLUS

For a two-phase load, the SORTOUT data set contains the output of the PRELOAD phase as DB2 row images for loading in the LOAD phase.

Note
You should not depend on the format of this proprietary data set; this format is subject to change without notice.

LOADPLUS uses this data set during the PRELOAD phase through the end of the LOAD phase. For most single-phase load jobs, LOADPLUS does not use the SORTOUT data set. However, for those single-phase jobs that do use it, LOADPLUS writes the data-sorting index keys and new DB2 row images to the SORTOUT data set during the COMBINED phase, but only reads from this data set if you restart your job. Refer to the following table for circumstances under which LOADPLUS requires or uses the SORTOUT data set:

Table 102: SORTOUT usage

<table>
<thead>
<tr>
<th>Load type</th>
<th>SORTOUT requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two-phase load</td>
<td>At least one required</td>
</tr>
<tr>
<td>Load type</td>
<td>SORTOUT requirement</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>■ Single-phase load</td>
<td>At least one required if you want to be able to restart</td>
</tr>
<tr>
<td>■ LOAD RESUME YES SHRLEVEL NONE or SHRLEVEL REFERENCE</td>
<td>Note the following considerations:</td>
</tr>
<tr>
<td>■ ORDER YES</td>
<td>■ If you do not specify a SORTOUT data set and the load job fails, you might have to recover the object. See “Recovering the DB2 object after terminating or canceling a LOADPLUS job” on page 478 for more information.</td>
</tr>
<tr>
<td>■ ORDER NO</td>
<td>■ This requirement does not apply if you are replacing one or more partitions, or replacing a table (specifying PART n REPLACE or TABLE tableName REPLACE).</td>
</tr>
<tr>
<td>■ Single-phase load</td>
<td>Not used</td>
</tr>
<tr>
<td>■ LOAD RESUME YES SHRLEVEL NONE</td>
<td></td>
</tr>
<tr>
<td>■ ORDER NO</td>
<td></td>
</tr>
<tr>
<td>■ Single-phase load</td>
<td>Not used</td>
</tr>
<tr>
<td>■ LOAD RESUME YES SHRLEVEL NONE or SHRLEVEL CHANGE</td>
<td></td>
</tr>
<tr>
<td>■ PART n REPLACE</td>
<td></td>
</tr>
<tr>
<td>■ Single-phase load</td>
<td>Not used</td>
</tr>
<tr>
<td>■ LOAD RESUME YES with TABLE tableName REPLACE</td>
<td></td>
</tr>
<tr>
<td>■ Single-phase load</td>
<td>Not used</td>
</tr>
<tr>
<td>■ LOAD REPLACE or LOAD RESUME NO</td>
<td></td>
</tr>
<tr>
<td>SQLAPPLY load</td>
<td>Not used</td>
</tr>
</tbody>
</table>

You can use either form of the data set name (SORTOUT or SORTOUT n). When specifying multiple data sets, you can use one or both forms. The variable n can be any valid single-digit alphanumeric or national character. To override the default ddname, use the LOADDN command option. If you plan to add more than one digit to the prefix or have LOADPLUS dynamically allocate more than nine SORTOUT data set names, you might have to recover the object.
data sets, you must override the default name to create a ddname prefix of six characters or less.

**Methods for allocating SORTOUT data sets**

You can use one of the following methods to allocate your SORTOUT data sets and determine the appropriate size for those data sets:

- **(recommended)** Have LOADPLUS dynamically allocate your SORTOUT data sets by using the DDTYPE command or installation option.
  
  For more information about dynamic allocation, see “Dynamic data set allocation in LOADPLUS on page 78 and “Dynamic allocation options” on page 393.

- Specify ANALYZE PAUSE or ANALYZE ONLY to have LOADPLUS provide an estimate of the space that is needed for your SORTOUT data sets.
  
  For more information about these options, see “ANALYZE option for estimating data set allocation in LOADPLUS” on page 465.

- Calculate the space allocation based on your knowledge of the table space that you are loading and the information in this section.

**Restrictions**

Note the following restrictions for allocating SORTOUT data sets:

- For a DSNUTILB load, you must have LOADPLUS dynamically allocate your SORTOUT data sets.

- Do not specify DD DUMMY or a DD specified with DSN=NULLFILE.

- To enable LOADPLUS to restart any time during the COMBINED phase or after the PRELOAD phase, SORTOUT cannot be a temporary data set. For information about the LOADPLUS treatment of temporary data sets, see “Work file validity and integrity checks in LOADPLUS” on page 85.

- For multiple load jobs running in a worklist environment, do not use SMS extended sequential data sets, including striped data sets, as SORTOUT data sets. When LOADPLUS first uses a data set, it defines the data set’s DCB attributes. With an SMS extended sequential data set, once these attributes are defined they cannot be changed. If this data set is reused, as with multiple load jobs in a worklist environment, it can require different DCB attribute values, such as a different block size or record length. LOADPLUS attempts to redefine these attributes if necessary. However, because MVS does not allow changes to these attributes after the file has been opened, LOADPLUS fails.
Dynamically allocating SORTOUT data sets

Note the following recommendations when dynamically allocating your SORTOUT data sets:

- If you plan to have LOADPLUS allocate more than nine load file (SORTOUT) data sets, use the LOADDN command or installation option to specify a ddname prefix that results in eight characters or less after LOADPLUS appends the highest data set number. For more information about specifying ddname prefixes, see “Specifying ddname prefixes” on page 81.

- For tables with variable length rows, LOADPLUS allocates the load (SORTOUT) data sets based on the average row length. If this calculation results in a large difference from the actual average, use the SIZEPCT option to adjust the file allocations.

Allocating multiple SORTOUT data sets in your JCL

LOADPLUS allows you to specify up to 16 SORTOUT data sets. To determine individual SORTOUT data set sizes when using multiple data sets, divide the total by the number of data sets.

LOADPLUS determines the optimal number of concurrent SORTOUT data sets to use based on such factors as the number of data sets that you specify in your DD statements, the number of sort processes that will fit in memory, and the number of partitions that are involved in the job. See “Using multiple SORTOUT data sets” on page 671 for details about how LOADPLUS optimizes the number of SORTOUT data sets and other performance implications of multitasking in LOADPLUS.

If you specify PRELOAD ANALYZE, LOADPLUS stops after PRELOAD optimization and displays a message that indicates the optimal number of SORTOUT (and SYSREC) data sets that LOADPLUS can process concurrently. You can use this information to adjust the number of data sets in your JCL. There is no advantage to specifying more SORTOUT data sets than the LOADPLUS optimizer selects.

For partitioned table spaces (except partition-by-growth), LOADPLUS uses SORTOUT DD statements up to the number of sort processes that LOADPLUS can run concurrently, based on available memory and other factors. For nonpartitioned and partition-by-growth table spaces, LOADPLUS uses only the first SORTOUT DD statement.

SORTWK data sets in LOADPLUS

SORTWK data sets are the work files that BMCSORT uses.
BMCSORT uses the data sets in the PRELOAD and LOAD phases for two-phase load, and in the COMBINED phase for single-phase processing.

For any job in which LOADPLUS performs a sort, you must allocate SORTWK files in one of the following ways:

- (recommended unless you are running a DSNUTILB load) Have BMCSORT dynamically allocate SORTWK data sets. For more information, see “Dynamically allocating SORTWK data sets” on page 447.

- Have LOADPLUS dynamically allocate SORTWK data sets (by specifying the DDTYPE option). For more information, see “Dynamically allocating SORTWK data sets” on page 447.
  
  This method is required for a DSNUTILB load.

- Explicitly specify SORTWK DD statements in your JCL. Use this option when you want to control the allocation of your SORTWK data sets. For more information, see “SORTWK data sets in LOADPLUS” on page 446 “Allocating SORTWK data sets in your JCL” on page 449.

**Dynamically allocating SORTWK data sets**

Dynamic allocation takes place when one of the following actions occurs:

- You specifically request dynamic allocation through command or installation options.

- BMCSORT determines that it needs more sort work space than other allocation methods provide (allocated in your JCL, dynamically allocated by LOADPLUS, or both).

Several factors affect this dynamic allocation as described in the following sections.

**LOADPLUS allocation (ACTIVE YES)**

**Note**

Unless you are running a DSNUTILB load, BMC recommends that you use BMCSORT to dynamically allocate your sort work data sets.

If LOADPLUS dynamic allocation is active for sort work data sets, LOADPLUS allocates at least 12 and up to 99 sort work data sets, depending on the total space required. LOADPLUS dynamic allocation is active when both of the following conditions exist:

- You specify ACTIVE YES for DDTYPE SORTWORK.
- Either you specify values for ENUMROWS, or ENUMROWS is automated.
Note the following considerations:

- File size estimation must be automated (AUTOENUMROWS=YES or ENUMROWS AUTO), or you must specify a value for ENUMROWS. Otherwise, LOADPLUS dynamic allocation is not active and, if BMCSORT allocation is active, BMCSORT performs dynamic allocation of sort work data sets.

- If you specify IFALLOC USE, LOADPLUS allocates sort work data sets, if needed, in addition to the ones specified in your JCL.

- When LOADPLUS dynamic allocation is active for sort work data sets, any values that you specify for SORTNUM, SORTDEVT, and the BMCSORT DYNALOC installation option are not used for allocation unless BMCSORT determines that it needs more sort work data sets than LOADPLUS allocated. In this case, BMCSORT uses these options, but only for allocation of the additional data sets.

- For tables with variable length rows, LOADPLUS allocates sort work (SORTWK) data sets based on the average row length. If this calculation results in a large difference from the actual average, use the SIZEPCT option to adjust the file allocations.

**BMCSORT allocation**

If BMCSORT allocation is active (see the following table), BMCSORT dynamically allocates your sort work files in the following cases:

- If LOADPLUS dynamic allocation is not enabled for sort work files (ACTIVE NO) and you have not specified SORTWK DD statements in your JCL

- In special cases when LOADPLUS dynamic allocation is enabled (see “Allocating SORTWK data sets in your JCL” on page 449)

- In special cases when you have allocated sort work data sets in your JCL (see “Allocating SORTWK data sets in your JCL” on page 449)

The following table describes how BMCSORT allocates sort work files based on values for the SORTDEVT and SORTNUM command options and the BMCSORT DYNALOC installation option:
Table 103: Factors that affect BMCSORT dynamic allocation of sort work data sets

<table>
<thead>
<tr>
<th>SORTDEVT and SORTNUM</th>
<th>Third parameter of BMCSORT DYNALOC</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORTDEVT specified</td>
<td></td>
<td>If you specify a SORTNUM value that is greater than 32, BMCSORT allocates the number of data sets that it determines it needs, up to the specified number of data sets per sort task. Otherwise, BMCSORT allocates the number of data sets that it determines it needs, up to 32 per sort task.</td>
</tr>
<tr>
<td>or</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SORTNUM n specified</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(where n is</td>
<td></td>
<td></td>
</tr>
<tr>
<td>greater than 0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ON or OFF</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If the value is OFF, specifying a value greater than 0 for SORTNUM or specifying SORTDEVT changes this value to ON.</td>
<td></td>
</tr>
<tr>
<td>SORTDEVT not</td>
<td></td>
<td>BMCSORT does not allocate any sort work data sets and attempts to perform sort processing in memory.</td>
</tr>
<tr>
<td>specified</td>
<td></td>
<td></td>
</tr>
<tr>
<td>and</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SORTNUM 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ON</td>
<td>BMCSORT allocates the number of data sets that it determines it needs, up to 32 per sort task.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>BMCSORT does not allocate any sort work data sets and attempts to perform sort processing in memory.</td>
</tr>
</tbody>
</table>

Allocating SORTWK data sets in your JCL

Unless you specify ACTIVE YES IFALLOC FREE for LOADPLUS dynamic allocation of sort work data sets, LOADPLUS uses any SORTWK DD statements that you specify in your JCL.

To allocate SORTWK data sets in your JCL, determine the space needed for these data sets by specifying ANALYZE PAUSE or ANALYZE ONLY. These options tell LOADPLUS to provide an estimate of the space that is needed for your work data sets. For more information, see “ANALYZE option for estimating data set allocation in LOADPLUS” on page 465.

When determining the number of concurrent tasks to run, LOADPLUS checks the amount of sort work space that is allocated. In this calculation, LOADPLUS uses only the primary allocation. The secondary allocation is not guaranteed.

Additional considerations

Note the following information when allocating SORTWK data sets in your JCL:

- You cannot allocate a SORTWK data set as any of the following data set types:
  - A VIO data set
  - A tape data set
  - A multi-volume data set
A data set in an SMS storage group that specifies EXTENDED FORMAT YES
BMCSORT does not support SORTWK data sets that extend beyond 65535
tracks on a single volume.

When you specify SORTWK DD statements in your JCL, BMCSORT dynamically
allocates additional sort work data sets when both of the following conditions
exist:

— BMCSORT determines that it needs more sort work space than you have
allocated.

— BMCSORT dynamic allocation is enabled (in other words, either the third
parameter of DYNALOC is ON, SORTDEVT is specified, or SORTNUM is
specified with a value greater than 0).

**SYSDISC data set in LOADPLUS**

The SYSDISC data set contains discarded input records. LOADPLUS requires a
SYSDISC data set if you want to save discarded records.

Records in the SYSDISC data set are those that are discarded when LOADPLUS
cannot load the record for any of the following reasons:

- An input value position is past the end of the input record.
- An input value is not entirely contained on the input record.
- A data conversion error (including a FIELDPROC error) occurs.
- A value generated for an identity column is out of range.
- A verification record does not match the table that is being loaded (for FORMAT
  BMCUNLOAD).
- LOADPLUS encounters an error byte during data translation.
- LOADPLUS encounters a substitution character during data translation, but you
  specified NOSUBS.
- An expansion error occurs during data translation.
- A VALIDPROC violation occurs.
- A record does not match any WHEN specification or table OBID.
- A record does not match any partition that is being loaded.
- A DB2 check constraint violation occurs.
- A DB2 referential constraint violation occurs.
- An informational referential constraint violation occurred (and you specified INFORI YES).
- A unique key violation occurs in the PRELOAD or COMBINED phase.
- An SQLCODE -330, -530, or -803 occurred during an SQLAPPLY load.
- A problem occurs with a referenced file used to load LOB or XML data.
- A problem occurs with an XML document.
- A conversion error occurs on a key that is derived from an expression.

LOADPLUS does not write discarded records to SYSDISC in the cases described in the following table:

<table>
<thead>
<tr>
<th>Condition</th>
<th>Type of discarded records not written to SYSDISC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input data set is a pipe</td>
<td>Records that meet any of the listed discard criteria</td>
</tr>
<tr>
<td>UNIQUECHECK NO specified for a two-phase load job</td>
<td>Duplicate records</td>
</tr>
<tr>
<td>UNIQUECHECK CLUSTER specified for a two-phase load job</td>
<td>Duplicates in non-data-sorting indexes</td>
</tr>
<tr>
<td>LOADPLUS evaluates the same input file for multiple INTO statements</td>
<td>Input records that are valid for at least one INTO statement, but not all INTO statements</td>
</tr>
</tbody>
</table>

**Note**

The number of discarded records in your SYSDISC data set might not match the number of discards that LOADPLUS reports because LOADPLUS issues a separate discard message for each violation. For example, if two input records are duplicates and they both violate a check constraint, LOADPLUS reports four discard violations, but the SYSDISC data set contains only two discarded input records.

LOADPLUS writes to this data set in the PRELOAD phase of a two-phase load or the COMBINED phase of a single-phase or SQLAPPLY load. To override the default ddname, use the DISCARDDN command option.

You can correct this discarded data and use it as input to a subsequent load job specifying RESUME YES.
**Note**

If you specified FORMAT BMCUNLOAD on the original load job, you must modify the discard file before you can use it as input to a subsequent load job. For instructions, contact BMC Customer Support.

LOADPLUS frees the SYSDISC data set at the end of the PRELOAD phase or the COMBINED phase. If you specify SYSDISC YES with the DELETEFILES option and no records were written to the SYSDISC file during the load process, LOADPLUS automatically deletes the SYSDISC file. LOADPLUS does not delete this data set if it contains discarded records. You must manually delete the SYSDISC data set in this case.

**Note**

For a DSNUTILB load, SYSDISC YES with the DELETEFILES option is ignored.

**Methods for allocating SYSDISC data sets**

You can use one of the following methods to allocate your SYSDISC data sets and determine the appropriate size for those data sets:

- Have LOADPLUS dynamically allocate your SYSDISC data sets by using the DDTYPE command or installation option.
  
  This method is required for a DSNUTILB load. For more information about dynamic allocation, see “Dynamic data set allocation in LOADPLUS” on page 78 and “Dynamic allocation options” on page 393.

- Specify ANALYZE PAUSE or ANALYZE ONLY to have LOADPLUS provide an estimate of the space that is needed for your SYSDISC data sets.
  
  For more information about these options, see “ANALYZE option for estimating data set allocation in LOADPLUS” on page 465.

- Calculate the space allocation based on your knowledge of the table space that you are loading and the information in this section.

**Additional considerations**

Note the following additional information about allocating SYSDISC data sets:

- Because LOADPLUS closes the SYSDISC data set at the end of the PRELOAD phase or the COMBINED phase, you should not specify FREE=CLOSE for SYSDISC data sets.

- To prevent your SYSDISC data sets from containing incompatible records from different load jobs, BMC recommends that you do not allocate your SYSDISC data sets with DISP=MOD.
■ Do not use SMS extended sequential data sets as SYSDISC data sets for multiple load jobs running in a worklist environment. When LOADPLUS first uses a data set, it defines the data set’s DCB attributes. With an SMS extended sequential data set, once these attributes are defined they cannot be changed. If this data set is reused, as with multiple load jobs in a worklist environment, it can require different DCB attribute values, such as a different block size or record length. LOADPLUS attempts to redefine these attributes if necessary. However, because MVS does not allow changes to these attributes after the file has been opened, LOADPLUS fails.

■ If you expect few or no errors, allocate a small amount of primary space and a large amount of secondary space.

■ When loading from an input data set that is in spanned-record format, LOADPLUS writes any discarded records to a discard data set that is in spanned-record format. If you have allocated a discard data set in your JCL, LOADPLUS overwrites the allocation parameters to RECFM=VBS, LRECL=32768.

**SYSERR data set in LOADPLUS**

The SYSERR data set contains information about errors and is for LOADPLUS internal use only. LOADPLUS always requires a SYSERR data set.

*Note*

You should not depend on the format of this proprietary data set; this format is subject to change without notice.

LOADPLUS uses this data set from the time LOADPLUS starts reading the SYSREC data set until the time LOADPLUS completes loading all of the tables. To override the default ddname for the SYSERR data set, use the ERRDDN command option.

*Note*

You can use the DISCARDS option to limit the number of SYSERR records that LOADPLUS generates.

**Methods for allocating SYSERR data sets**

You can use one of the following methods to allocate your SYSERR data sets and determine the appropriate size for those data sets:

■ Have LOADPLUS dynamically allocate your SYSERR data sets by using the DDTYPE command or installation option.

For more information about dynamic allocation, see “Dynamic data set allocation in LOADPLUS” on page 78 and “Dynamic allocation options” on page 393.
Specify ANALYZE PAUSE or ANALYZE ONLY to have LOADPLUS provide an estimate of the space that is needed for your SYSERR data sets.

For more information about these options, see “ANALYZE option for estimating data set allocation in LOADPLUS” on page 465.

Calculate the space allocation based on your knowledge of the table space that you are loading and the information in this section.

Additional considerations

Note the following additional information about allocating SYSERR data sets:

- For a DSNUTILB load, you must have LOADPLUS dynamically allocate your SYSERR data set.
- Do not specify DD DUMMY or a DD specified with DSN=NULLFILE.
- BMC recommends that you do not allocate a temporary data set for SYSERR. For information about the LOADPLUS treatment of temporary data sets, see “Work file validity and integrity checks in LOADPLUS” on page 85.
- For multiple load jobs running in a worklist environment, do not use SMS extended sequential data sets as SYSERR data sets. When LOADPLUS first uses a data set, it defines the data set’s DCB attributes. With an SMS extended sequential data set, once these attributes are defined they cannot be changed. If this data set is reused, as with multiple load jobs in a worklist environment, it can require different DCB attribute values, such as a different block size or record length. LOADPLUS attempts to redefine these attributes if necessary. However, because MVS does not allow changes to these attributes after the file has been opened, LOADPLUS fails.

SYSIN data sets

The SYSIN data set is an input data set that contains the utility command. The utility always requires the SYSIN data set.

The UTILINIT phase reads, parses, and verifies the utility command that is provided in this data set. The data set’s attributes must be specified as fixed length (RECFM is F, FB, or FBS), and the record length must be 80 columns (LRECL=80). The utility uses only columns 1 through 72.

SYSMAP data set in LOADPLUS

The SYSMAP data set contains mapping information between the table row identifier and the input record. LOADPLUS requires a SYSMAP data set only when invoking DSNUTILB.
You must have LOADPLUS dynamically allocate this data set. If you include a SYSMAP DD statement in your JCL, LOADPLUS ignores it.

**SYSPRINT data sets**

The SYSPRINT data set is the output data set that contains the utility's messages. The utility always requires the SYSPRINT data set.

*Note*  
You should not depend on the content and format of this data set (for example, as input to user-defined processes). Message content and format are subject to change without notice.

The utility overrides any data control block attributes that you specify in your JCL with DCB=(RECFM=VBA,LRECL=137,BLKSIZE=141).

Note the following considerations if you direct SYSPRINT to a tape or disk data set:

- The DSN messages from DB2 are lost.
- BMC does not recommend using the BUFNO parameter on the DD statement for this data set.

Note the following restrictions on SYSPRINT data sets in your DSNUTILB reorganizations:

- Do not specify FREE=CLOSE.
- You cannot use a PDS or PDSE.

For information about the level of messages that the utility displays and how to change the message level, see the description for the MSGLEVEL utility parameter.

**SYSPRIN2 data sets**

The SYSPRIN2 data set is an optional output data set that contains the utility's messages.

With the exceptions noted in this section, the content is identical to the content of the SYSPRINT data set.

SYSPRIN2 is not a substitute for SYSPRINT. If you include a SYSPRIN2 DD statement in your JCL, you must still include a SYSPRINT DD statement.
You should not depend on the content and format of this data set (for example, as input to user-defined processes). Message content and format are subject to change without notice.

The utility overrides any data control block attributes that you specify in your JCL with DCB=(RECFM=VBA,LRECL=137,BLKSIZE=141).

In a worklist environment, you can specify SYSPRIN2 DD SYSOUT=* to view output in real time from any of the BMC Utility products that run in that worklist.

Note the following considerations if you direct SYSPRIN2 to a tape or disk data set:

- The DSN messages from DB2 are lost.
- BMC does not recommend using the BUFNO parameter on the DD statement for this data set.

SYSPRIN2 data sets have the following restrictions:

- When you specify BMCSTATS YES or UPDATEDB2STATS YES, SYSPRIN2 does not contain the statistics report from the Common Statistics component.
- When invoking DSNUTILB, the utility ignores the SYSPRIN2 DD statement.

For information about the level of messages that the utility displays and how to change the message level, see the MSGLEVEL parameter description.

**SYSREC data sets in LOADPLUS**

SYSREC is an input data set that contains the input data that you are loading and, if applicable, the name of a file that contains LOB or XML data. LOADPLUS always requires at least one SYSREC data set.

LOADPLUS uses the data set in the PRELOAD phase for two-phase load or in the COMBINED phase for single-phase processing. At the end of the PRELOAD phase or the COMBINED phase, LOADPLUS frees this data set. However, you must specify a SYSREC DD statement in the JCL (which can be SYSREC DD DUMMY) or ensure that your LOAD command contains the INDSN option if you restart the job, regardless of the phase in which you restart.

You can use either form of the default data set name (SYSREC or SYSREC nn). For information about how to specify DD statements when using multiple data sets, see “Using multiple SYSREC data sets” on page 457. To override the default ddname, use the INDDN command option.
As an alternative to specifying SYSREC data sets in your JCL, you can have LOADPLUS dynamically allocate your SYSREC data sets by using the INDSN command option.

Requirements and restrictions

SYSREC data sets have the following general requirements and restrictions. For more restrictions when using multiple SYSREC data sets, see “Using multiple SYSREC data sets” on page 457.

- You must use a single SYSREC DD in the following cases:
  - When you specify ORDER PRESORTED
  - If you are generating values into an identity column and you want the values ordered in input order

- BMC strongly recommends that you do not use RECFM=FBS.

- If you are using RECFM=U, or have concatenated an uninitialized data set in your SYSREC DD (which results in RECFM=U), LOADPLUS processes only one record per block and produces unpredictable results.

- A data set that is in spanned-record format must be allocated as RECFM=YES with LRECL=32768.

- LOADPLUS does not support SYSREC data sets allocated as DSNTYPE=LARGE.

- Do not specify FREE=CLOSE for SYSREC data sets.

Using multiple SYSREC data sets

LOADPLUS permits up to 256 SYSREC DD statements, providing multitasking of the read process in the PRELOAD phase or in the COMBINED phase. When specifying multiple data sets, you can use one or both forms of the data set name (SYSREC or SYSRECnn).

Note

If LOADPLUS is dynamically allocating your input data sets and you need more than 99 data sets, use the INDDN command option to override the default data set name of SYSREC. Specify a ddname prefix that results in eight characters or less after LOADPLUS appends the highest data set number. For more information, see “Specifying ddname prefixes” on page 81.

LOADPLUS reads the SYSREC data sets in parallel only when loading data into a partitioned table space. For details about performance implications of multitasking in LOADPLUS, see “Enabling multitasking for performance in LOADPLUS” on page 670.
If you specify PRELOAD ANALYZE, LOADPLUS stops after the optimization process of the PRELOAD phase and displays a message that indicates the optimal number of sort and read tasks that it can process concurrently. You can use this information to adjust the number of SYSREC data sets in your JCL.

**Considerations**

The following considerations apply to multiple SYSREC data sets:

- Data in the SYSREC data sets does not have to be grouped by partition.
- Data for a particular partition can be in one or more SYSREC data sets.
- Although not required, BMC recommends that you order your SYSREC DD statements from the largest data set to the smallest to optimize performance.

**Requirements**

In addition to the general requirements and restrictions on SYSREC data sets ("Requirements and restrictions" on page 457), the following requirements apply to multiple SYSREC data sets:

- You can specify a mixture of DASD files and pipes as input. However, you cannot concatenate these files within one SYSREC DD.

- If you concatenate your input data sets, specify the data set with the largest block size and record length first.

---

**Note**

Alternatively, on the first data set in the concatenation, specify values for the BLKSIZE and LRECL parameters that will ensure that this data set has the largest block size and record length.

---

- If you use multiple pipe data sets, you must specify a separate SYSREC DD statement for each pipe data set, and the DCB parameters for all of the pipe data sets must be the same.

- Multiple SYSREC data sets must use the same record format (RECFM). In other words, they must all be allocated as either F or FB, or they must all be allocated as V, VB, or VBS. Mixing fixed and variable format data sets produces unpredictable results.

- Multiple SYSREC data sets that are defined with F or FB record format must use the same record length (LRECL).

**Specifying files that contain LOB or XML data**

In your SYSREC file, you can specify the name of a PDS, PDSE, or HFS file that contains your LOB or XML data. Note the following requirements:
If your referenced file is a PDS or PDSE, you must include a member name in the name provided in the input file.

If your referenced file is an HFS file, your input file must specify the full path name of the file, as in the following example:

/home/rdajdm/lobfr1

SYSUT1 data sets in LOADPLUS

The index work data set contains index key entries and becomes the input for the index-building process.

For a two-phase load, LOADPLUS uses this data set in the PRELOAD phase and in the LOAD phase. When used with a single-phase LOAD RESUME YES SHRLEVEL NONE job, LOADPLUS writes the index key entries to the index work data set during the COMBINED phase, but only reads from this data set if you restart your job. The following table describes the circumstances under which LOADPLUS requires this data set:

Table 105: SYSUT1nn usage

<table>
<thead>
<tr>
<th>Load type</th>
<th>SYSUT1nn requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two-phase load</td>
<td>At least one required if any indexes are participating in the load</td>
</tr>
<tr>
<td>Single-phase load</td>
<td>For LOB and XML loads, at least one required</td>
</tr>
<tr>
<td>LOAD RESUME YES SHRLEVEL NONE or SHRLEVEL REFERENCE</td>
<td>For all other jobs, at least one required if you want to be able to restart and any indexes are participating in the load</td>
</tr>
<tr>
<td>Without TABLE tableName REPLACE</td>
<td>Note: If you do not specify a SYSUT1 data set and the load job fails, you might have to recover the object. See “Recovering the DB2 object after terminating or canceling a LOADPLUS job” on page 478 for more information.</td>
</tr>
<tr>
<td>Single-phase load</td>
<td>Not used</td>
</tr>
<tr>
<td>LOAD RESUME YES with TABLE tableName REPLACE</td>
<td></td>
</tr>
<tr>
<td>Single-phase load</td>
<td>Not used</td>
</tr>
<tr>
<td>LOAD REPLACE or LOAD RESUME NO</td>
<td></td>
</tr>
<tr>
<td>SQLAPPLY load</td>
<td>Not used</td>
</tr>
</tbody>
</table>

Chapter 4 Building and executing LOADPLUS jobs 459
You can always run with only one SYSUT1 data set. However, if you want to improve performance, you can specify more than one data set. For more information about specifying multiple SYSUT1 data sets, see “Allocating SYSUT1 data sets in your JCL” on page 461.

To override the default ddname or ddname prefix SYSUT1, use the WORKDDN command option.

**Methods for allocating SYSUT1 data sets**

You can use one of the following methods to allocate your SYSUT1 data sets and determine the appropriate size for those data sets:

- (recommended) Have LOADPLUS dynamically allocate your SYSUT1 data sets by using the DDTYPE command or installation option.
  
  For more information about dynamic allocation, see “Dynamic data set allocation in LOADPLUS” on page 78 and “Dynamic allocation options” on page 393.

- Specify ANALYZE PAUSE or ANALYZE ONLY to have LOADPLUS provide an estimate of the space that is needed for your SYSUT1 data sets.
  
  For more information about these options, see “ANALYZE option for estimating data set allocation in LOADPLUS” on page 465.

- Calculate the space allocation based on your knowledge of the table space that you are loading and the information in this section.

**Restrictions**

Note the following restrictions for allocating SYSUT1 data sets:

- For a DSNUTILB load, you must have LOADPLUS dynamically allocate your SYSUT1 data sets.

- Do not specify DD DUMMY or a DD specified with DSN=NULLFILE.

- To enable LOADPLUS to restart any time after the PRELOAD or COMBINED phase has begun, the SYSUT1 cannot be a temporary data set.

- For multiple load jobs running in a worklist environment, do not use SMS extended sequential data sets, including striped data sets, as SYSUT1 data sets. When LOADPLUS first uses a data set, it defines the data set’s DCB attributes. With an SMS extended sequential data set, once these attributes are defined they cannot be changed. If this data set is reused, as with multiple load jobs in a worklist environment, it can require different DCB attribute values, such as a different block size or record length. LOADPLUS attempts to redefine these attributes if necessary. However, because MVS does not allow changes to these attributes once the file has been opened, LOADPLUS fails.
Dynamically allocating SYSUT1 data sets

Note the following recommendations when dynamically allocating SYSUT1 data sets:

- If you plan to have LOADPLUS allocate more than 99 index work file (SYSUT1) data sets, use the WORKDDN command or installation option to specify a ddname prefix that results in eight characters or less after LOADPLUS appends the highest data set number. For more information about specifying ddname prefixes, see “Specifying ddname prefixes” on page 81.

- If you are loading a table space with a large number of indexes, BMC recommends that you specify a single SYSUT1 data set to avoid data set allocation limitations of the operating system.

Allocating SYSUT1 data sets in your JCL

Note the following recommendations when allocating SYSUT1 data sets in your JCL.

Specifying a single work data set

If you want a single work data set for all non-data-sorting indexes, specify one SYSUT1 DD statement.

Specifying multiple work data sets

There is no penalty for specifying more SYSUT1 data sets than LOADPLUS requires. Using multiple work data sets reduces the amount of DASD that is required for loading if the key lengths vary a great deal. Also, if enough sort work space and memory are available, using multiple work data sets allows LOADPLUS to build nonclustering indexes concurrently. For more information about performance and DASD considerations, see “SYSUT1 data sets” on page 657.

If you specify more than one SYSUT1 data set, specify a DD statement named SYSUT1nn for each data set. Specifying nn creates a unique ddname; the nn has no relation to the index name. Use the following table to determine how many data sets to specify based on the indexes that exist on your table space:

Table 106: Number of SYSUT1 data sets to allocate

<table>
<thead>
<tr>
<th>ORDER value</th>
<th>Clustering index status</th>
<th>SYSUT1 data sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Any</td>
<td>1 per index</td>
</tr>
<tr>
<td>ORDER value</td>
<td>Clustering index status</td>
<td>SYSUT1 data sets</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>YES</td>
<td>Clustering index is not partitioned</td>
<td>1 per index</td>
</tr>
<tr>
<td></td>
<td>Clustering index is partitioned</td>
<td>(1 per index) - 1</td>
</tr>
<tr>
<td></td>
<td>No clustering index, but a partitioned partitioning index exists on the table space</td>
<td>(1 per index) - 1</td>
</tr>
<tr>
<td></td>
<td>No clustering index and no partitioned partitioning index</td>
<td>1 per index</td>
</tr>
</tbody>
</table>

Note the following additional considerations:

- Ensure that you include any LOB or XML indexes, including implicitly-defined indexes, in your calculations.

- If the number of SYSUT1 DD statements is greater than one and less than the number of participating non-data-sorting indexes, LOADPLUS terminates.

- If you are loading a table space with a large number of indexes, BMC recommends that you specify a single SYSUT1 data set to avoid data set allocation limitations of the operating system.

When using multiple SYSUT1 data sets, LOADPLUS assigns the index with the largest key length to the first SYSUT1 DD statement in the JCL, the index with the second-largest key length to the second SYSUT1 DD statement, and so on. This assignment of indexes to data set by key length can assist you in allocating the sizes of your data sets and allows you to place data sets on different devices. Due to this assignment, you should allocate the space for the participating index with the largest key length in the first SYSUT1 DD statement in your JCL, space for the index with the second-largest key length in the second SYSUT1 DD statement, and so on.

### SYSIDCIN data set in LOADPLUS

SYSIDCIN is the input data set that contains your IDCAMS command statements.

LOADPLUS uses these statements to delete and redefine user-defined (VCAT-defined) data sets or the staging data sets for your VCAT-defined data sets. LOADPLUS issues the commands that the SYSIDCIN data set contains. You are responsible for command specifications and results.

LOADPLUS requires a SYSIDCIN data set if the value of the REDEFINE option is YES and either of the following conditions exists:

- You want to delete and redefine your existing VCAT-defined VSAM data sets as part of the load.
You specify a load job that uses staging data sets and you are loading VCAT-defined data sets.

If you specify REDEFINE YES and do not provide a SYSIDCIN data set, LOADPLUS operates as if you specified REDEFINE NO.

**Note**

You still have the option of deleting and redefining VCAT-defined data sets when you specify PRELOAD PAUSE. However, when you specify PRELOAD CONTINUE or PRELOAD LOAD, the only way to delete and redefine VCAT-defined data sets as part of the load is to specify REDEFINE YES and provide a SYSIDCIN data set that contains the necessary IDCAMS command statements.

If any referenced object is VCAT-defined, the UTILINIT phase reads, parses, and performs minimal verification checks on the commands in this data set. LOADPLUS issues the commands during load processing.

SYSIDCIN can be either a single or concatenated list of sequential data sets, partitioned data set members, or both. You must define this data set as fixed length with blocked records (RECFM=FB), and the record length must be 80 bytes (LRECL=80). LOADPLUS uses only columns 1 through 72. To override the default ddname, SYSIDCIN, use the IDCDDN command option.

For important restart information, see “Restarting LOADPLUS” on page 467.

**Guidelines for providing IDCAMS commands in the SYSIDCIN data set**

LOADPLUS supports the following IDCAMS commands and their parameters as defined in the IBM *DFSMS Access Method Services for Catalogs* document:

- DELETE
- DEFINE
- SET
- IF-THEN-ELSE command sequence
- DO/END (with restrictions)

LOADPLUS does not allow the following IDCAMS command specifications. LOADPLUS checks for these specifications during the UTILINIT phase and terminates with an error message if it finds any of them.

- An IF statement without a DEFINE, DELETE, or SET command
- A nested IF statement
- Different data set names specified in an IF statement
- More than one DO/END statement within a single THEN or ELSE clause
- A DO/END statement existing outside of an IF statement
An embedded comment (a comment within a comment)
A comment on a DO statement that continues to the next line
A keyword that continues to the next line

To avoid a failure during processing, ensure that each DEFINE has a corresponding DELETE specified before it and that each DELETE has a corresponding DEFINE specified after it. Ensure that both commands specify the same data set name.

You can use the SET command to reset the IDCAMS condition code if a failure occurs in the LOAD or COMBINED phase and you do not want LOADPLUS to terminate.

When you specify SHRLEVEL NONE

Using the guidelines in the preceding section, specify commands only for those VCAT-defined data sets that you want to delete and redefine. LOADPLUS reuses any data sets participating in the load that have no corresponding IDCAMS commands. LOADPLUS ignores any command that references a data set that does not participate in the load and issues message BMC50604I. LOADPLUS ignores any empty SYSIDCIN data set and continues processing.

When you specify SHRLEVEL CHANGE (except an SQLAPPLY load) or SHRLEVEL REFERENCE

Using the guidelines in “Guidelines for providing IDCAMS commands in the SYSIDCIN data set” on page 463, specify commands that operate only on the staging data sets for your VCAT-defined data sets. Always use the naming conventions outlined in “Staging data sets” on page 98.

To define the staging data sets and avoid definition errors, provide a DELETE statement followed by a SET MAXCC=0 command (in case the delete process fails) before each DEFINE statement. LOADPLUS processes this set of statements (DELETE, SET MAXCC, DEFINE) for each object as the LOAD or COMBINED phase begins for that object. LOADPLUS ignores any command that references one of the original VCAT-defined data sets and issues message BMC50604I.

UTPRINT data sets

The utility always requires the UTPRINT data set if sorting is necessary.

The presence of this data set tells the utility to report sort messages. However, the actual messages for each sort process appear in separate SYSnnnn data sets, where nnnnnn is a system-assigned sequential number. You cannot specify a sequential or partitioned data set for UTPRINT. UTPRINT supports only SYSOUT data sets.
WARNING

JES3 users should be aware of a limitation within JES3 that does not allow concurrent tasks to share SYSOUT data sets. This limitation means that you cannot use additional sort routine reporting DDs (other than UTPRINT) if they are defined as JES3 SYSOUT data sets and when the utility is multitasking its sort activity. If you attempt to use an unsupported DD, you risk S1FB abends when concurrent sort tasks are running.

Other data sets

Because it uses BMCSORT, the utility ignores any traditional sort routine DD statements (such as $SORTPARM and DFSPARM) that you specify.

ANALYZE option for estimating data set allocation in LOADPLUS

If you specify ANALYZE PAUSE or ANALYZE ONLY, LOADPLUS gathers information about the table space and the index space.

In addition to cardinality and average row size, the ANALYZE phase provides estimated data set sizes for the following data sets:

- Load (SORTOUT)
- Work (SYSUT1)
- Sort work (SORTWK)
- Discard (SYSDISC)
- Error (SYSERR)
- Image copy (BMCCPY, BMCCPZ, BMCCPY, and BMCCCY)

Note

ANALYZE does not provide estimates for LOB or XML copy data sets. These data sets must be dynamically allocated.

LOADPLUS writes these statistics to SYSPRINT.

To use the ANALYZE function, you must also either enable automated file size estimation or specify an appropriate numeric value for the ENUMROWS command option.

ANALYZE PAUSE or ANALYZE ONLY provides the estimated information in table format. Messages BMC51530I, BMC51531I, and BMC51532I provide the heading information and multiple BMC51533I messages provide the estimates. LOADPLUS
issues a separate BMC51533I message for each data set and provides the following information:

- Data set name
- Number of kilobytes
- Primary and secondary 3380 cylinder quantities
- Primary and secondary 3390 cylinder quantities
- Index name, where applicable

Note
For rows that contain VARCHAR columns or tables that contain EDITPROCs, ANALYZE ONLY or ANALYZE PAUSE might report a secondary quantity for SORTOUT that is larger than the reported primary quantity because LOADPLUS bases the primary quantity on the average row length and the secondary quantity on the maximum row length from the DB2 catalog. In this instance, BMC recommends that you provide a secondary quantity of approximately 25 percent of the primary quantity.

As an alternative to using ANALYZE to help you estimate sizes for data sets, you can have LOADPLUS dynamically allocate your data sets for you. To use dynamic allocation, you must

- Either enable automated file size estimation (AUTOENUMROWS=ON or ENUMROWS AUTO), or specify an appropriate numeric value for the ENUMROWS command option.

- Enable dynamic data set allocation, either in your installation options or with the DDTYPE command option.

For more information about dynamic allocation, see “Dynamic data set allocation in LOADPLUS” on page 78 and “Dynamic allocation options” on page 393.

If you do not use the PAUSE or ONLY keywords with ANALYZE, LOADPLUS also gathers the information described in this section. However, instead of pausing or stopping, LOADPLUS continues processing. If dynamic allocation is enabled, LOADPLUS uses the ANALYZE phase information to dynamically allocate your data sets. In this case, the ANALYZE phase does not write the statistics to SYSPRINT.

Running LOADPLUS jobs

After you have built your LOADPLUS job, the next step is to run the job.
This section describes how to invoke a job, considerations for restarting it, and how to terminate or cancel it. This section also provides recovery steps in case you cannot restart your job.

Invoking LOADPLUS

You normally invoke LOADPLUS as a batch job by specifying execution of the module AMUUMAIN on the EXEC statement of your JCL and including the required EXEC statement parameters.

You must also specify any DD statements that LOADPLUS requires, as described in preceding sections.

Ensure that all required libraries are available and APF-authorized as described in “LOADPLUS STEPLIB DD statement” on page 431.

Restarting LOADPLUS

With the exceptions described in the sections that follow, you can restart LOADPLUS from any phase.

In general, when you encounter a failure, you can correct the problem and restart the load with either the RESTART or RESTART(PHASE) parameter on your EXEC statement. LOADPLUS issues messages as it loads and rebuilds each DB2 object. The BMCSYNC table contains an entry for each DB2 object that is involved in the load and its current status. The BMCUTIL table contains an entry for each load job and its current status.

LOADPLUS relies on the object structure and input data to remain unchanged between the original run and the restarted run:

- **Input data**: You must use the same input data that you used originally and it must be in the same order. For example, if you are loading concatenated data sets, these data sets must be in the same order as they were originally, and you must not include any additional data sets. Similarly, if LOADPLUS encountered duplicates during the original job, do not remove them before you restart. Data that is missing or that is in a different order in the restarted job than in the original job produces unpredictable results.

- **Tables**: You must load the same table that you specified in the original run. The table structure must not be changed (as with an ALTER) before restarting the load.
- **Indexes**: The same indexes must participate in the restarted run that participated in the original run. Additionally, the structure of those indexes must not be changed before restarting the load.

The following sections describe information that you need to know when restarting or recovering from specific situations or when a particular environment, type of load job, or object is involved. Use the following table to help locate this information:

<table>
<thead>
<tr>
<th>Environment or object</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVE option</td>
<td>“Dynamic allocation” on page 470</td>
</tr>
<tr>
<td>Copy data sets</td>
<td>“Copy data sets” on page 469</td>
</tr>
<tr>
<td>Data sharing environment</td>
<td>“Data sharing environment” on page 469</td>
</tr>
<tr>
<td>DEFINE NO objects</td>
<td>“DEFINE NO objects” on page 469</td>
</tr>
<tr>
<td>DELETEFILES process</td>
<td>“DELETEFILES” on page 470</td>
</tr>
<tr>
<td>DSNUTILB load</td>
<td>“DSNUTILB load” on page 470</td>
</tr>
<tr>
<td>Duplicate unique keys</td>
<td>“Duplicate unique keys” on page 470</td>
</tr>
<tr>
<td>Dynamic allocation</td>
<td>“Dynamic allocation” on page 470</td>
</tr>
<tr>
<td>ENFORCE option</td>
<td>“ENFORCE option” on page 471</td>
</tr>
<tr>
<td>ENUMROWS option</td>
<td>“ENUMROWS option” on page 471</td>
</tr>
<tr>
<td>FASTSWITCH option</td>
<td>“FASTSWITCH option” on page 471</td>
</tr>
<tr>
<td>Field specification</td>
<td>“Field specification” on page 471</td>
</tr>
<tr>
<td>IDCAMS DEFINE process</td>
<td>“IDCAMS DEFINE process” on page 471</td>
</tr>
<tr>
<td>Inadequate space failure, DB2 object</td>
<td>“Inadequate space failure on a DB2 object during a LOAD REPLACE job” on page 471</td>
</tr>
<tr>
<td>Inadequate space failure, SYSERR data set</td>
<td>“Inadequate space failure on SYSERR data set” on page 472</td>
</tr>
<tr>
<td>INDEX UPDATE option</td>
<td>“INDEX UPDATE” on page 473</td>
</tr>
<tr>
<td>Inline image copies</td>
<td>“Inline image copies” on page 474</td>
</tr>
<tr>
<td>Nonparticipating nonpartitioned indexes</td>
<td>“Nonparticipating nonpartitioned indexes (SKIPIX option)” on page 474</td>
</tr>
<tr>
<td>ORDER option</td>
<td>“ORDER option” on page 474</td>
</tr>
<tr>
<td>Partition number</td>
<td>“Partition number” on page 474</td>
</tr>
<tr>
<td>Partition-by-growth table spaces</td>
<td>“Partition-by-growth table spaces” on page 474</td>
</tr>
<tr>
<td>Real-time statistics</td>
<td>“Real-time statistics” on page 475</td>
</tr>
<tr>
<td>Referential constraint checking</td>
<td>“Referential constraint checking” on page 475</td>
</tr>
</tbody>
</table>
Copy data sets

You should not specify DISP=MOD for image copy data sets because LOADPLUS does not reset the data set to empty but appends to any data already present in the file.

Data sharing environment

On restart in a data sharing environment, LOADPLUS can use either the same member that was used in the original load or any other member in the group.

DEFINE NO objects

For objects that are defined with DEFINE NO, if LOADPLUS is unable to materialize a data set that is associated with the tables that you are loading, LOADPLUS attempts to materialize the underlying data sets for all of the tables before terminating.

If you are loading a table whose table space or index spaces are defined with DEFINE NO and there is a check constraint or VALIDPROC on the table,
LOADPLUS attempts to materialize the underlying data sets. However, if LOADPLUS cannot resolve the constraint or if the VALIDPROC prevents LOADPLUS from materializing the data sets, LOADPLUS terminates. To load the table in this situation, you must manually materialize the associated data sets before you restart the job.

**DELETEFILES**

To restart during DELETEFILES processing, specify RESTART without (PHASE).

**DSNUTILB load**

After the UTILINIT phase, LOADPLUS passes the restart parameter to DSNUTILB. Restart processing is handled by the IBM DB2 LOAD utility.

**Duplicate unique keys**

If you are restarting because the job terminated during discard processing while handling duplicate unique keys, LOADPLUS issues message BMC51449W if any indexes are left in a RECOVER pending status. In this case, you can recover the affected indexes by running RECOVER PLUS or the IBM RECOVER utility.

**Dynamic allocation**

On restart, LOADPLUS automatically reallocates dynamically allocated data sets.

Changing any dynamic data set allocation option on restart such that it results in different ddnames or a different number of DDs than the original option can produce an error. If you need to change the number of SORTOUT or SYSUT1 work files, resubmit the job with a parameter of NEW.

In addition, you cannot change the value for the following options, which are related to dynamic allocation, on any restart:

- ACTIVE
- ENUMROWS

To change the value of other dynamic data set allocation options, specify RESTART(PHASE).

If you run out of space when dynamically allocating copy data sets, you can perform one of the following procedures:

**To change COPY YES to COPY NO**

1 Specify COPY NO.
2 Specify the appropriate COPYPEND option.

3 Restart your job.

To allocate copy data sets in your JCL

1 Delete and reallocate the data sets.

2 Specify DD statements for the data sets in your JCL.

3 Change your syntax to IFALLOC USE for the data sets.

4 Restart your job.

**ENFORCE option**

You cannot change the value of the ENFORCE option on restart.

**ENUMROWS option**

You cannot change the value of the ENUMROWS option on restart.

**FASTSWITCH option**

You cannot change the value of the FASTSWITCH option on restart.

**Field specification**

You cannot change your field specification on restart.

**IDCAMS DEFINE process**

If you are restarting because a failure occurred during IDCAMS DEFINE processing, you must manually define the data set.

**Inadequate space failure on a DB2 object during a LOAD REPLACE job**

Failure during the LOAD phase can result in an unusable table space. If a failure occurs on a LOAD REPLACE or LOAD RESUME YES PART n REPLACE job due to inadequate space for a DB2 object, complete the actions that the following table describes to help eliminate the cause of the failure:
### Table 108: Reallocation process for inadequate space failure

<table>
<thead>
<tr>
<th>Data set definition</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>The table space or index data sets are VCAT-defined and no SYSIDCIN DD statement is provided.</td>
<td>Allocate new data sets for the data sets that failed, specifying a larger primary or secondary space or both.</td>
</tr>
<tr>
<td>The table space or index data sets are VCAT-defined and you included a SYSIDCIN DD statement for redefining them as part of the load.</td>
<td>Either specify REDEFINE NO and allocate new data sets for the data sets that caused the failure, or alter the primary and secondary space quantities that are specified in the SYSIDCIN data set.</td>
</tr>
<tr>
<td>The table space or index data sets are STOGROUP-defined.</td>
<td>Specify REDEFINE YES and alter the table space and index primary and secondary quantity values.</td>
</tr>
<tr>
<td>The table space is partition-by-growth and you specify or default to SHRLEVEL NONE.</td>
<td>Alter the table space to decrease the PCTFREE or FREEPAGE value. For more information, see “Partition-by-growth table spaces” on page 474.</td>
</tr>
</tbody>
</table>

### Restrictions

The following restrictions apply when reallocating after an inadequate space failure:

- After the PRELOAD phase, LOADPLUS does not use changes to FREEPAGE, PCTFREE, or PIECESIZE values. LOADPLUS uses other values, such as storage group definitions and data set passwords, if they have changed.

- You cannot correct this problem by altering the primary and secondary quantity values to -1 (thus requesting that secondary extents be created by using a sliding-scale method). LOADPLUS ignores this change on restart.

After correcting the problem that caused the inadequate space failure, restart the load job with the RESTART option. LOADPLUS does not reload or rebuild DB2 objects that were already loaded or built successfully. However, if you reallocate any data sets that were successfully loaded or built, restart the load process at the beginning of the LOAD phase with RESTART(PHASE).

**WARNING**

If you reallocate data sets that were successfully loaded or built in the COMBINED phase, you will lose the data contained in those data sets.

### Inadequate space failure on SYSERR data set

When allocating a SYSERR data set, underestimating its size can result in an x37 abend, indicating an out-of-space condition. Perform one of the following set of steps to recover and restart your job.
Dynamically allocated SYSERR data sets

Because LOADPLUS verifies your SYSERR data set on restart, you must copy the data from the original data set to the new SYSERR data set. However, if you are dynamically allocating your SYSERR data set, you must use the same data set name as the original, so you need a temporary data set to store the data until you can allocate a data set with the same name as the original.

Perform the following steps if you are dynamically allocating your SYSERR data set:

1. Copy the data from your original data set to a temporary data set.
2. Delete the original data set.
3. Allocate a larger SYSERR data set with the same data set name and DCB parameters as the original. These might not be the same DCB parameters that were coded in your JCL.
4. Copy the data from the temporary data set to the new SYSERR data set.
5. Restart your job.

SYSERR data sets that are allocated in your JCL

Perform the following steps if you are allocating your SYSERR data set in your JCL:

1. Allocate a larger SYSERR data set and make sure that you use the same DCB parameters as your original data set. These might not be the same DCB parameters that were coded in your JCL.
2. Because LOADPLUS verifies your SYSERR data set on restart, you must copy the data from the original data set to the new data set.
3. Restart your job.

INDEX UPDATE

If you specified INDEX UPDATE and a failure occurred during index update processing, you must recover your table space and indexes. If you attempt to restart, LOADPLUS terminates and issues message BMC51435S.

When you specify INDEX UPDATE and index update processing completes, but the job fails in the LOAD or COMBINED phase, you must restart the job by specifying RESTART without PHASE. If you specify RESTART(PHASE), LOADPLUS terminates and issues message BMC51436E.
**Inline image copies**

If you want an inline image copy and you restart your two-phase job after the load job terminates in the LOAD phase, you must specify RESTART(PHASE).

If you are restarting a two-phase load job and do not specify RESTART(PHASE) or you are restarting a single-phase load job, LOADPLUS changes the value of the INLINE command option to NO, regardless of the value that you specified for the INLINE command or the INLINECP installation option.

**Nonparticipating nonpartitioned indexes (SKIPIX option)**

If you change the SKIPIX option on restart after the load job terminated in any phase other than UTILINIT, LOADPLUS issues message BMC50113E and terminates.

**ORDER option**

You cannot change the value of the ORDER command option on restart.

**Partition number**

You cannot change the partition number on restart.

**Partition-by-growth table spaces**

The following restrictions apply when you restart a load of a partition-by-growth table space:

- If the value for MAXPARTITIONS changes between the time the job fails and the time you restart it, LOADPLUS issues message BMC50221U and terminates.

- If DB2 adds partitions before you restart a job, LOADPLUS issues message BMC50221U and terminates.

- If a SHRLEVEL NONE load job fails because insufficient space is available, BMC recommends that you perform an ALTER TABLESPACE to decrease the PCTFREE or FREEPAGE value before restarting the job. However, LOADPLUS only honors these changes in the following situations:
  - For a single-phase load, LOADPLUS honors these changes for those partitions to which it has not started loading.
  - For a two-phase load, LOADPLUS honors these changes only if the failure occurs during the PRELOAD phase.
Real-time statistics

Unlike the IBM DB2 LOAD utility, LOADPLUS attempts to update DB2 real-time statistics tables any time you restart your load job. If LOADPLUS is unable to update one or more columns, LOADPLUS issues message BMC50290I and continues processing.

Referential constraint checking

If you restart a job that checks referential constraints, LOADPLUS might not be able to start at the beginning of the job and, therefore, might not be able to check all referential constraints. In this case, LOADPLUS sets CHKP status on the table space.

RESUME YES SHRLEVEL NONE

If your load job failed in any phase other than UTILINIT or UTILTERM, and you specify LOAD RESUME YES with SHRLEVEL NONE in effect, ensure that the participating objects are stopped before you restart your job.

Secondary extents

On a restart, if you alter the primary and secondary quantity values to -1 (thus requesting that secondary extents be created by using a sliding-scale method), LOADPLUS ignores this change.

Single-phase load

In general, the restart parameter RESTART(PHASE) works the same as RESTART when you restart a single-phase load job. For more criteria that might affect which RESTART option to use, refer to the other topics in this section.

For a single-phase LOAD RESUME YES SHRLEVEL NONE load job, you cannot restart the job when either of the following conditions exists:

- You did not allocate a SYSUT1 data set (or specify dynamic allocation for SYSUT1) on the original job. This restriction does not apply if you are replacing a table (specifying TABLE tableName REPLACE).

- You specified ORDER YES and did not allocate a SORTOUT data set (or specify dynamic allocation for SORTOUT) on the original job. This restriction does not apply if you are replacing one or more partitions, or replacing a table (specifying PART n REPLACE or TABLE tableName REPLACE).

See “Recovering the DB2 object after terminating or canceling a LOADPLUS job” on page 478 for information about recovering your load job.
SORTOUT, SYSUT1, and SYSERR data sets

If you specified DISP=NEW for the SORTOUT, SYSUT1, or SYSERR data sets, you must change your JCL to DISP=OLD before restarting. Alternatively, you can allocate the data sets with DISP=MOD, or allocate the data sets in a preceding step and then specify DISP=OLD for the data sets in the load step. For more information, see “Check for data set attributes” on page 85.

Do not delete your SORTOUT, SYSUT1, or SYSERR data sets before restarting your job. Doing so produces unpredictable results.

SQLAPPLY load

As with other LOADPLUS jobs, if you restart an SQLAPPLY job, you must use the same input data that you used originally and it must be in the same order. This is particularly critical for an SQLAPPLY job because of the way that High-speed Apply handles restart processing.

In general, the restart parameter RESTART(PHASE) works the same as RESTART when you restart an SQLAPPLY job. For more criteria that might affect which RESTART option to use, refer to the other topics in this section.

Statistics

The following considerations apply to restarted jobs when you specify BMCSTATS YES or UPDATEDB2STATS YES:

- On restart, LOADPLUS does not update statistics if, in the original job, any participating table space partitions were completely loaded or any participating index partitions were completely updated or built.

- You can change the TSSAMPLEPCT option on restart.

Note

For restart information about real-time statistics, see “Real-time statistics” on page 475.

SWITCHTIME option

If the SWITCHTIME process did not start before the load job failed, you can change the SWITCHTIME option on restart. If the SWITCHTIME process started before the load job failed, LOADPLUS immediately performs the rename or FASTSWITCH process on restart.
SYSDISC data sets

To prevent data loss, BMC makes the following recommendations regarding SYSDISC data sets and restart:

■ Do not delete your SYSDISC data set before restarting your job.
■ If you need to reallocate your SYSDISC data set due to an out-of-space condition (x37 abend), copy the contents of your original file to the new file before restarting your job.

Two-phase load

In general, the restart option that you specify to restart a two-phase load job depends on whether you want to restart from the last restart sync point or at the beginning of the last incomplete phase:

■ To restart from the last restart sync point, specify RESTART without (PHASE). LOADPLUS takes restart sync points as each phase completes and as the processing of each DB2 object completes.

  Note
  The SYNC option (on the LOAD command) controls the frequency with which rows are updated in the BMCSYNC table. It does not control the sync points that LOADPLUS uses when restarting.

■ To restart at the beginning of the last incomplete phase, specify RESTART(PHASE). The utility ID must exist in the BMCUTIL table.

Terminating or canceling a LOADPLUS job

You can stop the LOADPLUS job in one of the following ways:

■ If you want to end the utility immediately and want the ability to restart your job, cancel the job by using the MVS or TSO CANCEL command.

■ If you want to end the job and do not intend to restart, terminate the load job by performing one of the following actions. If the job is currently running, it terminates at the next sync point.

  — Specify TERM on the restart parameter of the utility.

  — Delete the corresponding rows from the BMCUTIL, BMCSYNC, and BMCDICT tables.
Recovering the DB2 object after terminating or canceling a LOADPLUS job

You must perform certain recovery steps when any of the following conditions exists:

- You terminated the job by
  - Specifying the TERM parameter on the EXEC statement
  - Deleting the row from the BMCUTIL table that corresponds to the utility job
- The job ended for a reason other than you terminating it, and you do not intend to restart it (or you cannot restart it due to omitted SORTOUT or SYSUT1 data sets).
- You canceled the job, and you do not intend to restart it (or you cannot restart it due to omitted SORTOUT or SYSUT1 data sets).

The following table describes the recovery steps that you need to take.

Note
If you canceled your LOADPLUS job, or the job ended for a reason other than you terminating it, you can usually restart the job. When you restart the job, you do not normally need to perform any additional recovery steps. However, for information about special restart conditions, see “Restarting LOADPLUS” on page 467.

Table 109: Recovering after termination

<table>
<thead>
<tr>
<th>Phase in which job terminated</th>
<th>Specified options</th>
<th>Other conditions</th>
<th>Is object usable?</th>
<th>Recovery steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any phase other than UTILTERM</td>
<td>SHRLEVEL or LOAD REPLACE SHRLEVEL CHANGE or LOAD RESUME YES SHRLEVEL CHANGE PART n REPLACE</td>
<td>The object is usable (with original data).</td>
<td>No recovery steps are required.</td>
<td></td>
</tr>
<tr>
<td>UTILINIT</td>
<td>Any</td>
<td>The object is usable.</td>
<td>No recovery steps are required.</td>
<td></td>
</tr>
<tr>
<td>Phase in which job terminated</td>
<td>Specified options</td>
<td>Other conditions</td>
<td>Is object usable?</td>
<td>Recovery steps</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------</td>
<td>-----------------</td>
<td>------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>DSNUTILB</td>
<td>Any</td>
<td></td>
<td>The object might be usable.</td>
<td>For information about any recovery required, see the appropriate IBM DB2 documentation.</td>
</tr>
<tr>
<td>PRELOAD</td>
<td>LOAD REPLACE</td>
<td></td>
<td>The object is usable.</td>
<td>No recovery steps are required.</td>
</tr>
<tr>
<td>PRELOAD</td>
<td>Any option other than LOAD REPLACE</td>
<td></td>
<td>The object is usable.</td>
<td>Start the table space and index spaces.</td>
</tr>
<tr>
<td>LOAD</td>
<td>LOAD RESUME YES SHRLEVEL NONE with one of the following specifications:</td>
<td>No PART specifications and no TABLE tableName REPLACE specification</td>
<td>The table space contains data before the load job.</td>
<td>Complete the following steps:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Start the table space and index spaces.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 Recover the table space.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3 Recover the indexes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4 Rerun the job.</td>
</tr>
<tr>
<td>Phase in which job terminated</td>
<td>Specified options</td>
<td>Other conditions</td>
<td>Is object usable?</td>
<td>Recovery steps</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------</td>
<td>-----------------</td>
<td>------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>LOAD</td>
<td>LOAD RESUME YES SHRLEVEL NONE PART n REPLACE</td>
<td>The object is <em>not</em> usable.</td>
<td>Complete the following steps:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 Start the table and index spaces.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 Recover the partition.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 Recover the corresponding partitions of the partitioned indexes and all nonpartitioned indexes.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4 Rerun the job.</td>
<td></td>
</tr>
<tr>
<td>LOAD or COMBINED</td>
<td>LOAD RESUME YES INTO TABLE tableName REPLACE</td>
<td>The object is <em>not</em> usable.</td>
<td>Complete the following steps:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 Start the table and index spaces.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 Either rerun the job or recover the table space and then recover the indexes.</td>
<td></td>
</tr>
<tr>
<td>LOAD or COMBINED</td>
<td>LOAD REPLACE SHRLEVEL NONE</td>
<td>The object is <em>not</em> usable.</td>
<td>Complete the following steps:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 Start the table and index spaces.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 Either rerun the job or recover the table space and then recover the indexes.</td>
<td></td>
</tr>
<tr>
<td>Phase in which job terminated</td>
<td>Specified options</td>
<td>Other conditions</td>
<td>Is object usable?</td>
<td>Recovery steps</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------</td>
<td>-------------------------------------------------------</td>
<td>-------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>COMBINED</td>
<td>LOAD RESUME YES SHRLEVEL NONE, single-phase</td>
<td></td>
<td>The object is <em>not</em> usable.</td>
<td>Complete the following steps:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Start the table and index spaces.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 Recover the table space and then recover the indexes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3 Rerun the job.</td>
</tr>
<tr>
<td>LOAD or COMBINED</td>
<td>SHRLEVEL NONE or LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY or LOAD RESUME YES SHRLEVEL CHANGE without PART n REPLACE</td>
<td>The table space has been loaded and all discard processing has completed.</td>
<td>It might be only the indexes that are <em>not</em> usable.</td>
<td>Complete the following steps:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Determine whether the table space was loaded by checking message BMC51475I or by querying the BMCSYNC table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 Start the table and index spaces.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3 Recover the indexes. (You must perform this last step before you can use any of the loaded data.)</td>
</tr>
<tr>
<td>Phase in which job terminated</td>
<td>Specified options</td>
<td>Other conditions</td>
<td>Is object usable?</td>
<td>Recovery steps</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------</td>
<td>-----------------</td>
<td>------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>UTILTERM</td>
<td>SHRLEVEL NONE</td>
<td></td>
<td>The object is usable.</td>
<td>No recovery steps are required.</td>
</tr>
<tr>
<td></td>
<td>or LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>or LOAD RESUME YES SHRLEVEL CHANGE without PART n REPLACE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UTILTERM</td>
<td>SHRLEVEL REFERENCE</td>
<td></td>
<td>The rename or FASTSWITCH process started but did not complete.</td>
<td>Manually complete or back out the rename or FASTSWITCH process and remove pending statuses. <strong>Note:</strong> BMC strongly recommends that you restart your job rather than attempt to rename the staging data sets manually.</td>
</tr>
<tr>
<td></td>
<td>or LOAD REPLACE SHRLEVEL CHANGE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>or LOAD RESUME YES SHRLEVEL CHANGE PART n REPLACE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Examples of LOADPLUS jobs

This chapter presents several examples of jobs that were run by using the LOADPLUS product.

Overview of LOADPLUS examples

This chapter presents several examples of jobs that were run by using the LOADPLUS product.

Each example includes the following information:

- A description of the load job
- The LOADPLUS job stream
- The SYSPRINT from the job

Some examples show additional information that might be useful to understand a particular feature or to see the results of the load.

All of these examples have the following common properties:

- These examples use a single BMC product load library, assuming that this library contains the following files and libraries:
  - LOADPLUS load files
  - Common component load files that LOADPLUS requires
  - SAS/C transient library (required for statistics processing)

- When data sets are allocated in the JCL, these examples use minimal data set space allocations.

- The value for the FILECHK installation option is set to WARN. This value prevents the jobs from failing when they encounter temporary data sets.

You can find copies of the JCL for these examples in members AMUEXnn in the HLQ.LLQCNTL installation data set (where nn is the number of the example, HLQ
is the high-level qualifier set during installation, and \textit{LLQ} is the low-level qualifier or prefix set during installation).

For syntax details, see “Syntax of the \textbf{LOAD} command” on page 163. For information about JCL statement requirements, see “Building and executing \textbf{LOADPLUS} jobs” on page 425.

Use the following table to locate an example with a specific load or object type, or one that uses a specific option. Use the chapter table of contents to find the example.

\textbf{Table 110: Cross-reference of examples by function}

<table>
<thead>
<tr>
<th>Function</th>
<th>Relevant examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Load type</strong></td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>1, 2, 4-6, 8, 11, 13-15, 17, 18, 19, 21</td>
</tr>
<tr>
<td>RESUME YES</td>
<td>3, 12, 22</td>
</tr>
<tr>
<td>RESUME YES PART \textit{n} REPLACE</td>
<td>9, 20</td>
</tr>
<tr>
<td>RESUME YES TABLE \textit{tableName} REPLACE</td>
<td>16</td>
</tr>
<tr>
<td>RESUME NO</td>
<td>7, 10</td>
</tr>
<tr>
<td>Single phase</td>
<td>1, 3, 13, 14, 16, 18-22</td>
</tr>
<tr>
<td>Two phase</td>
<td>1, 2, 4-11, 15, 17</td>
</tr>
<tr>
<td>CSV input</td>
<td>13</td>
</tr>
<tr>
<td>UNLOAD PLUS input</td>
<td>14</td>
</tr>
<tr>
<td>XML input</td>
<td>21</td>
</tr>
<tr>
<td>LOB input</td>
<td>21</td>
</tr>
<tr>
<td>DSNUTILB load</td>
<td>22</td>
</tr>
<tr>
<td>SQLAPPLY load</td>
<td>12</td>
</tr>
<tr>
<td>User exit</td>
<td>5</td>
</tr>
<tr>
<td><strong>Object type</strong></td>
<td></td>
</tr>
<tr>
<td>Partitioned table space</td>
<td>2, 3, 5, 8, 9, 11, 12, 14, 15, 20-22</td>
</tr>
<tr>
<td>Partition-by-growth table space</td>
<td>10</td>
</tr>
<tr>
<td>Segmented table space</td>
<td>1, 4, 6, 7, 13, 16-19</td>
</tr>
<tr>
<td>Multi-table table space</td>
<td>4, 16, 19</td>
</tr>
<tr>
<td>Compressed table space</td>
<td>8, 9, 16, 18, 20, 21</td>
</tr>
<tr>
<td>Unicode table space</td>
<td>17</td>
</tr>
<tr>
<td>Clustering index, unique</td>
<td>2-4, 7-13, 15, 18, 19</td>
</tr>
<tr>
<td>Clustering index, non-unique</td>
<td>5, 16, 21</td>
</tr>
<tr>
<td>Function</td>
<td>Relevant examples</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Nonclustering index, unique</td>
<td>6, 15, 16, 18, 21</td>
</tr>
<tr>
<td>Nonclustering index, non-unique</td>
<td>2-5, 8, 9, 11, 14, 15, 18-20</td>
</tr>
<tr>
<td>Partitioning, nonclustering index, non-unique</td>
<td>14</td>
</tr>
<tr>
<td>Index with random ordered keys</td>
<td>22</td>
</tr>
<tr>
<td>ASUSRPRT data set</td>
<td>8</td>
</tr>
<tr>
<td>ASUSRPRT DD DUMMY</td>
<td>11, 20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Command option</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVE NO</td>
<td>3, 8</td>
</tr>
<tr>
<td>ACTIVE YES</td>
<td>11, 16, 19-22</td>
</tr>
<tr>
<td>ANALYZE PAUSE</td>
<td>8</td>
</tr>
<tr>
<td>AND/OR condition</td>
<td>4</td>
</tr>
<tr>
<td>BLOBF</td>
<td>21</td>
</tr>
<tr>
<td>BMCSTATS YES</td>
<td>2, 11</td>
</tr>
<tr>
<td>CCSID</td>
<td>17, 21</td>
</tr>
<tr>
<td>CENTURY</td>
<td>8</td>
</tr>
<tr>
<td>CLOBF</td>
<td>21</td>
</tr>
<tr>
<td>COPYDDN</td>
<td>3, 9</td>
</tr>
<tr>
<td>COPY NO</td>
<td>7, 10, 13, 15, 22</td>
</tr>
<tr>
<td>COPY YES</td>
<td>2-5, 8, 9, 11, 16, 19-21</td>
</tr>
<tr>
<td>COPYPEND NO</td>
<td>13, 22</td>
</tr>
<tr>
<td>DDTYPE</td>
<td>3, 8, 11, 12, 16, 19-22</td>
</tr>
<tr>
<td>DEFAULTIF</td>
<td>5, 6</td>
</tr>
<tr>
<td>DELETEFILES</td>
<td>12, 14, 16, 20</td>
</tr>
<tr>
<td>DISCARDYN DD</td>
<td>3, 9</td>
</tr>
<tr>
<td>DISCARDS</td>
<td>7, 10</td>
</tr>
<tr>
<td>DSNPAT</td>
<td>11, 12, 21, 22</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>17, 21</td>
</tr>
<tr>
<td>ENFORCE CHECK CONSTRAINTS</td>
<td>10</td>
</tr>
<tr>
<td>(ENFORCE=CHECK specified in installation options)</td>
<td></td>
</tr>
<tr>
<td>ENFORCE CONSTRAINTS</td>
<td>18</td>
</tr>
<tr>
<td>ENUMROWS</td>
<td>10, 12, 14, 19</td>
</tr>
<tr>
<td>Function</td>
<td>Relevant examples</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>ERRDDN</td>
<td>3, 9</td>
</tr>
<tr>
<td>EXIT</td>
<td>5</td>
</tr>
<tr>
<td>FASTSWITCH</td>
<td>19</td>
</tr>
<tr>
<td>FORMAT BMCUNLOAD</td>
<td>14</td>
</tr>
<tr>
<td>FORMAT CSV</td>
<td>13</td>
</tr>
<tr>
<td>IFALLOC</td>
<td>11</td>
</tr>
<tr>
<td>IFDISCARDS</td>
<td>7</td>
</tr>
<tr>
<td>IGNORE</td>
<td>7, 10</td>
</tr>
<tr>
<td>INDDN</td>
<td>3, 9, 20-22</td>
</tr>
<tr>
<td>INDEX UPDATE</td>
<td>9</td>
</tr>
<tr>
<td>INDSN</td>
<td>2, 14</td>
</tr>
<tr>
<td>INLINE NO</td>
<td>5, 8, 9, 11, 16, 19, 20</td>
</tr>
<tr>
<td>INTO TABLE (multiple)</td>
<td>4, 6, 9, 19</td>
</tr>
<tr>
<td>INTO TABLE (one)</td>
<td>1-3, 5, 7, 8, 10-15, 17, 18, 20-22</td>
</tr>
<tr>
<td>INTO TABLE REPLACE</td>
<td>16</td>
</tr>
<tr>
<td>LOADDN</td>
<td>3, 9</td>
</tr>
<tr>
<td>LOG NO</td>
<td>2-4, 8, 9, 11, 15</td>
</tr>
<tr>
<td>NULLIF</td>
<td>5, 7, 10, 13, 20, 22</td>
</tr>
<tr>
<td>ORDER NO</td>
<td>3, 6, 9, 11, 12</td>
</tr>
<tr>
<td>ORDER YES</td>
<td>2, 4, 5, 8, 12, 15, 16, 18-20</td>
</tr>
<tr>
<td>PART</td>
<td>2, 9</td>
</tr>
<tr>
<td>PART REPLACE</td>
<td>9, 20</td>
</tr>
<tr>
<td>POSITION ( start: end)</td>
<td>2, 3, 7-12, 15-17, 19-22</td>
</tr>
<tr>
<td>POSITION ( start)</td>
<td>1, 5-7, 10, 22</td>
</tr>
<tr>
<td>POSITION (*)</td>
<td>4, 7, 10, 13, 22</td>
</tr>
<tr>
<td>PREFORMAT</td>
<td>19</td>
</tr>
<tr>
<td>PRELOAD CONTINUE</td>
<td>1, 2, 4, 5-11, 15, 17</td>
</tr>
<tr>
<td>RECOVERYDDN</td>
<td>3, 9</td>
</tr>
<tr>
<td>REDEFINE NO</td>
<td>2, 8, 11</td>
</tr>
<tr>
<td>REDEFINE YES</td>
<td>6</td>
</tr>
<tr>
<td>REGISTER ALL</td>
<td>16, 19, 20</td>
</tr>
<tr>
<td>REGISTER ddname</td>
<td>5</td>
</tr>
</tbody>
</table>
Example 1 LOAD REPLACE of a segmented table space

This example illustrates three runs of a LOAD REPLACE of a segmented table space with no indexes and shows the use of data conversions.
This example runs the following three jobs. With the exception of the changes to the LOAD command statement, the JCL is the same for all jobs.

- SYSPRINT A ("Output from a two-phase load (example 1A)" on page 490) shows the output when the job is run as a two-phase load. PRELOAD CONTINUE is specified in the JCL to override the PRELOAD=LOAD default from the installation options module.

- In the second case, the PRELOAD CONTINUE option is removed from the LOAD command statement. The PRELOAD=LOAD default value tells LOADPLUS to use single-phase processing. SYSPRINT B ("Output from a single-phase load (example 1B)" on page 493) shows the single-phase load output.

- In the third version of this job, the SHRLEVEL REFERENCE option is added to the command statement, overriding the SHRLEVEL=(NONE,NONE) installation option value, to improve the accessibility of the data. SYSPRINT C ("Output from a LOAD REPLACE SHRLEVEL REFERENCE load (example 1C)" on page 495) shows the SHRLEVEL REFERENCE load output.

In this example, LOADPLUS loads the data into a segmented, STOGROUP-defined table space that contains only one table with no indexes. LOADPLUS replaces all existing rows and deletes and redefines the VSAM data set for the space.

No sort work data sets are allocated because there are no indexes and LOADPLUS is not sorting the data.

The following data conversions take place:

- Character data to small integer format
- Timestamp in character format to DB2 timestamp format
- Date in character format to DB2 date format

Because no COPY option is included on the LOAD command, LOADPLUS does not make a copy. Because the COPYPEND installation option is set to YES, LOADPLUS places the table space in COPY pending status and the job completes with return code 4.

The SYSPRINT output indicates that conversion errors occurred. Because the EXEC statement designates MSGLEVEL(1), LOADPLUS issues message BMC51501E to identify the erroneous input fields.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

Note that the JCL defines the SORTOUT and SYSERR data sets as DISP=(MOD,CATLG,CATLG) and the SYSDISC data set as...
DISP=(NEW,CATLG,CATLG). The EXEC statement includes the NEW/RESTART restart parameter.

The following figure shows the JCL for example 1A:

```
//EXAMPL1A EXEC PGM=AMUUMAIN,
//         PARM='SSID,EXAMPLE1,NEW/RESTART,,MSGLEVEL(1),AMU$OPTO'
//               SSID,UTILID,RSTART,NOTIFY,MSGLEVEL
//STEPLIB DD DISP=SHR,DSN=product.libraries
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//         DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD SYSOUT=* 
//SYSSORT DD SYSOUT=* 
//SYSSORT DD SYSOUT=* 
//SYSDUMP DD SYSOUT=* 
//SYSSSDUMP DD SYSOUT=* 
//SYSCORE DD DISP=SHR,DSN=AMU.VQA.EXAMPL01.SYSREC
//SORTOUT DD DSN=AMU.EXAMPL01A.SORTOUT,
//         UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
//SYSSYSENDD DS N=AMU.EXAMPL01A.SYSDISC,
//         UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
//* 
//SYSIN DD * 
LOAD REPLACE 
INTO TABLE EXAMPLE1.TBL1 
(SMINT_COL1 POSITION(1) CHARACTER(3) 
,TIMESTAMP_COL2 POSITION(4) TIMESTAMP EXTERNAL(19) 
,DATE_COL3 POSITION(23) DATE EXTERNAL 
,VARCHAR_COL4 POSITION(33) CHARACTER(40) TRIM 
,CHAR_COL5 POSITION(73) CHARACTER(7) ) 
/* 
```

The following figure shows the change to the SYSIN for example 1B. (PRELOAD CONTINUE is removed.)

```
//SYSIN DD * 
LOAD REPLACE 
INTO TABLE EXAMPLE1.TBL1 
(SMINT_COL1 POSITION(1) CHARACTER(3) 
,TIMESTAMP_COL2 POSITION(4) TIMESTAMP EXTERNAL(19) 
,DATE_COL3 POSITION(23) DATE EXTERNAL 
,VARCHAR_COL4 POSITION(33) CHARACTER(40) TRIM 
,CHAR_COL5 POSITION(73) CHARACTER(7) ) 
/* 
```

The following figure shows the change to the SYSIN for example 1C.

```
//SYSIN DD * 
LOAD REPLACE 
SHRLEVEL REFERENCE 
INTO TABLE EXAMPLE1.TBL1 
(SMINT_COL1 POSITION(1) CHARACTER(3) 
,TIMESTAMP_COL2 POSITION(4) TIMESTAMP EXTERNAL(19) 
,DATE_COL3 POSITION(23) DATE EXTERNAL 
,VARCHAR_COL4 POSITION(33) CHARACTER(40) TRIM 
,CHAR_COL5 POSITION(73) CHARACTER(7) ) 
/* 
```
### Example 1 LOAD REPLACE of a segmented table space

#### Output from a two-phase load (example 1A)

The following figure shows the SYSPRINT output for example 1A:

```sql
<table>
<thead>
<tr>
<th>LOADPLUS for DB2 Reference Manual</th>
</tr>
</thead>
</table>
```

Example 1 LOAD REPLACE of a segmented table space

#### Result

```sql
Example 1 LOAD REPLACE of a segmented table space

```

---

**Example 1 LOAD REPLACE of a segmented table space**

The following figure shows the SYSPRINT output for example 1A:

```
<table>
<thead>
<tr>
<th>LOADPLUS for DB2 Reference Manual</th>
</tr>
</thead>
</table>
```
Example 1 LOAD REPLACE of a segmented table space

**LOADPLUS for DB2 Reference Manual**
The following figure shows the SYSPRINT output for example 1B:

**Example 1 LOAD REPLACE of a segmented table space**

---

**Output from a single-phase load (example 1B)**

---

**Chapter 5 Examples of LOADPLUS jobs**

493
Output from a LOAD REPLACE SHLEVEL REFERENCE load (example 1C)

The following figure shows the SYSPRINT output for example 1C:
Example 1 LOAD REPLACE of a segmented table space
Example 1 LOAD REPLACE of a segmented table space

### Example 1 LOAD REPLACE of a segmented table space

Example 1 LOAD REPLACE of a segmented table space

Chapter 5  Examples of LOADPLUS jobs  497
Example 1 LOAD REPLACE of a segmented table space

LOADPLUS for DB2 Reference Manual
Example 2 LOAD REPLACE of a partitioned table space

This example shows a LOAD REPLACE of a table space with 10 partitions, a unique clustering index, and a non-unique nonpartitioned index.

Only the nonpartitioned index is processed by using a SYSUT1 work file, so the JCL includes only one SYSUT1 DD statement. Eight of ten partitions are loaded in this example by specifying a range of partitions.

The LOAD command includes the INDSN option, which tells LOADPLUS to dynamically allocate the input data set. Notice that the JCL does not include a SYSREC DD statement.

Because the LOAD command includes the ORDER YES option, LOADPLUS sorts the data rows and partitioned index keys. Because the LOAD command does not include the UNIQUECHECK NO option, LOADPLUS defaults to UNIQUECHECK YES, which tells LOADPLUS to check the indexes for uniqueness in the PRELOAD phase.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

The default value of INLINECP=YES and the command option COPY YES REGISTER NONE creates inline copies but does not register them with DB2. DDTYPE LOCPFCPY ACTIVE NO disables dynamic allocation for the primary copy data set, overriding the default in the installation options module. The copy DD statements in the JCL direct LOADPLUS to create a single copy of each partition, with each copy in a separate data set.

LOADPLUS updates statistics in both the DB2 catalog and the DASD MANAGER PLUS database statistics tables. Because no ASUSRPRT DD was included in the JCL, the statistics display in the LOADPLUS SYSPRINT.

The following figure shows the JCL for example 2:

```
//EXAMPLE2 EXEC PGM=AMUUMAIN,  
  /* PARM='&SSID,EXAMPLE2,NEW/RESTART,,MSGLEVEL(1),AMU$OPTO'  
    ** SSHD,UTILID ,RSTART,NOTIFY,MSGLEVEL  
//STEPLIB DD DISP=SHR,DSN=product.libraries  
// DD DISP=SHR,DSN=DB2.DSNEXIT  
// DD DISP=SHR,DSN=DB2.DSNLOAD  
//SYSPRINT DD SYSOUT=*  
//SYSOUT DD SYSOUT=*  
//UTPRINT DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
//**  
//SORTOUT DD DSN=AMU.EXAMP002.SORTOUT.
```
Example 2 LOAD REPLACE of a partitioned table space
SYSPRINT for example 2

The following figure shows the SYSPRINT output for example 2.

Note

To reduce the length of the output in this book, the figure shows one set of statistics for each object type. Ellipses (...) at the beginning of the line indicate the omitted statistics sections.
Example 2 LOAD REPLACE of a partitioned table space

LOADPLUS for DB2 Reference Manual
Example 2 LOAD REPLACE of a partitioned table space

Chapter 5 Examples of LOADPLUS jobs 503
Example 2 LOAD REPLACE of a partitioned table space

LOADPLUS for DB2 Reference Manual
Example 2 LOAD REPLACE of a partitioned table space

Example 2 LOAD REPLACE of a partitioned table space

BMC50482I 0: LOAD COMPLETE. ELAPSED TIME = 00:00:00 DSN = 'DEHJCAT.DSNDBD.AMUEX2DB.IX0.I0001.A007'
BMC50477I 7: PARTITION = 7, ROWS/KEYS = 0, I/O WAITS = 5, DDNAME = SYS00029
BMC50482I 7: BUILD COMPLETE. ELAPSED TIME = 00:00:00 DSN = 'DEHJCAT.DSNDBD.AMUEX2DB.IX0.I0001.A007'
BMC50477I 6: PARTITION = 6, ROWS/KEYS = 0, I/O WAITS = 5, DDNAME = SYS00028
BMC50482I 6: LOAD COMPLETE. ELAPSED TIME = 00:00:00 DSN = 'DEHJCAT.DSNDBD.AMUEX2DB.IX0.I0001.A006'
BMC50477I 5: PARTITION = 5, ROWS/KEYS = 0, I/O WAITS = 5, DDNAME = SYS00027
BMC50482I 5: BUILD COMPLETE. ELAPSED TIME = 00:00:00 DSN = 'DEHJCAT.DSNDBD.AMUEX2DB.IX0.I0001.A005'
BMC50477I 4: PARTITION = 4, ROWS/KEYS = 0, I/O WAITS = 7, DDNAME = SYS00026
BMC50482I 4: LOAD COMPLETE. ELAPSED TIME = 00:00:00 DSN = 'DEHJCAT.DSNDBD.AMUEX2DB.EXT01.I0001.A004'
BMC50477I 3: PARTITION = 3, ROWS/KEYS = 7, I/O WAITS = 5, DDNAME = SYS00025
BMC50482I 3: BUILD COMPLETE. ELAPSED TIME = 00:00:00 DSN = 'DEHJCAT.DSNDBD.AMUEX2DB.EXT01.I0001.A003'
BMC50477I 2: PARTITION = 2, ROWS/KEYS = 0, I/O WAITS = 0, DDNAME = SYS00024
BMC50482I 2: LOAD COMPLETE. ELAPSED TIME = 00:00:00 DSN = 'DEHJCAT.DSNDBD.AMUEX2DB.EXT01.I0001.A002'
BMC50477I 1: PARTITION = 1, ROWS/KEYS = 60, I/O WAITS = 7, DDNAME = SYS00023
BMC50482I 1: BUILD COMPLETE. ELAPSED TIME = 00:00:00 DSN = 'DEHJCAT.DSNDBD.AMUEX2DB.EXT01.I0001.A001'
BMC50482I 0: LOAD COMPLETE. ELAPSED TIME = 00:00:00 DSN = 'DEHJCAT.DSNDBD.AMUEX2DB.EXT01.I0001.A000'
BMC50477I 0: PARTITION = 0, ROWS/KEYS = 0, I/O WAITS = 0, DDNAME = SYS00022

Example 2 LOAD REPLACE of a partitioned table space

LOAD STATISTICS: 835 KEYs LOADED INTO INDEX 'EXAMPLE2.IX0 PARTITION 10'

LOAD STATISTICS: 835 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT05'

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT01' PARTITION 10

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT01' PARTITION 7

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT01' PARTITION 6

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT01' PARTITION 4

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT01' PARTITION 3

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT01' PARTITION 2

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT01' PARTITION 1

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT01' PARTITION 0

LOAD PHASE COMPLETE. ELAPSED TIME = 00:00:00

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT07'

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT06'

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT05'

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT04'

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT03'

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT02'

LOAD STATISTICS: 0 KEYs LOADED INTO TABLE SPACE 'AMUEX2DB.EXT01'

LOAD PHASE COMPLETE. ELAPSED TIME = 00:00:00

LOAD PHASE COMPLETE. ELAPSED TIME = 00:00:00
Example 2 LOAD REPLACE of a partitioned table space

EXTENTS ---------------------- 1  VOLCOUNT ---------------------- 1  DVTYPE ------------------- 3390  VOLUME ------------------ DEVS01

COLUMN ------------------------- DATE_OF_SALE
ASSOCIATIONS ------------------ AMUEX2DB.EX2TS
TABLE ------------------------- EXAMPLE2.TBL1
ATTRIBUTES
COLNO ------------------------- 9  COLTYPE ----------------- DATE  LENGTH ------------------ 4  MULLS ------------------- N
SCALE -------------------------- 0
STATISTICS
COLLARD ------------------------ 11  COLLUMS -------------- 0  COLAVG ------------------ 4  COLLIN ------------------- 4
HIGH2K ------------------------ X'19820105'  HIGH2K (CHAR) --------- X'19820105'  LOW2K -------------- X'19820101'  LOW2K (CHAR) ---------- X'19820101'

MOST FREQUENT VALUES
FREQUENCY ---------------------- 0.39521  COLVAL---------------- X'19820617'
FREQUENCY ---------------------- 0.11976  COLVAL---------------- X'19900126'
FREQUENCY ---------------------- 0.11976  COLVAL---------------- X'19820706'
FREQUENCY ---------------------- 0.118563 COLVAL---------------- X'19820706'
FREQUENCY ----------------------- 0.092216 COLVAL---------------- X'19820810'
FREQUENCY ----------------------- 0.070659 COLVAL---------------- X'19830101'
FREQUENCY ----------------------- 0.070659 COLVAL---------------- X'19961012'
FREQUENCY ----------------------- 0.030593 COLVAL---------------- X'19860910'

LENCSTATS v11.02.00 REPORT FOR DEHJ v1115 (111)       TIME 2015-03-09-13.57.28.989262                                          6

KEYCARD ------------------------ DATE_OF_SALE
ASSOCIATIONS ------------------ AMUEX2DB.EX2TS
TABLE ------------------------- EXAMPLE2.TBL1
ATTRIBUTES
COLLARD ------------------------ 9  COLTYPE ----------------- DATE  LENGTH ------------------ 4  MULLS ------------------- N
STATISTICS
COLLARD ------------------------ 5  COLLUMS -------------- 0  COLAVG ------------------ 4  COLLIN ------------------- 4

COLUMN ------------------------- DATE_OF_SALE PART 001
ASSOCIATIONS ------------------ AMUEX2DB.EX2TS
TABLE ------------------------- EXAMPLE2.TBL1
ATTRIBUTES
COLLARD ------------------------ 5  COLTYPE ----------------- DATE  LENGTH ------------------ 4  MULLS ------------------- N

COLUMN ------------------------- DATE_OF_SALE PART 002
ASSOCIATIONS ------------------ AMUEX2DB.EX2TS
TABLE ------------------------- EXAMPLE2.TBL1
ATTRIBUTES
COLLARD ------------------------ 5  COLTYPE ----------------- DATE  LENGTH ------------------ 4  MULLS ------------------- N

INDEX ------------------------- EXAMPLE2.IXI
ASSOCIATIONS ------------------ AMUEX2DB.EX2TS
TABLE ------------------------- EXAMPLE2.TBL1
HISTORY
SAVESTATS --------------------- Y  UPDATEDBZ -------------- L  UTILCODE ------------------ LOCATION ------------------ DEHJ
SAMPLING ---------------------- N
ATTRIBUTES
PIECESIZE --------------------- 2097152  TYPE TS --------------- (BLANK)  UNIQUEULE ----------- 0  KEYLENGTH -------------- 10
SPARSE ------------------------- N
INDEXSPACE ------------------- IX1  PGSIZE --------------- 4  EITYPE ------------------ (BLANK)  COMPRESS ----------- N
COLNAME ----------------------- SELLER_ID
STATISTICS
FIRSTKEYCARD ------------------ 2  FULLKEYCARD --------- 2  NACTIVE ------------- 6  TBCHORD -----------------.R85
NUMNONLEAF ------------------- 2  CLUSTERING ---------- 2  NLAF ------------------ 2  NODEF ------------------ 2
AVGKEYLEN --------------------- 10  REINDEXSPACE ------- 15  REINDEXSPACE KB ------------ 720  REINDEXLEVELS ---- 2
HIGH2K ------------------------ X'19820705'  HIGH2K (CHAR) --------- X'19820705'  LOW2K -------------- X'19820701'  LOW2K (CHAR) ---------- X'19820701'

DATAREFAT --------------------- 31  ALLOCATION
SPACE -------------- 15  SPACE KB ----------------- 720  NUMDATASETS ------------ 1

INDEXPART ---------------------- EXAMPLE2.IX1
ASSOCIATIONS ------------------ AMUEX2DB.EX2TS
TABLE ------------------------- EXAMPLE2.TBL1
HISTORY
SAVESTATS --------------------- Y  UPDATEDBZ -------------- L  UTILCODE ------------------ LOCATION ------------------ (NULL)
SAMPLING ---------------------- N
ATTRIBUTES
DOTY -------------------------- 1  SOTY --------------- 1  PCTFREE ------------- 0  FREEPAGE ----------- 0
ALLCOC ------------------------ 1  STORYTEP --------- 1  NUCLEKELEN ----------- 10  VCATNAME ------------- DENSCAT
RBA_FORMAT ------------------- EXTENDED  STORNAME ------------ AMUGAST
STATISTICS
FIRSTKEYCARD ------------------ 2  FULLKEYCARD --------- 2  NACTIVE ------------- 6  TBCHORD -----------------.R85
NUMNONLEAF ------------------- 2  CLUSTERING ---------- 2  NLAF ------------------ 2  NODEF ------------------ 2
AVGKEYLEN --------------------- 10  REINDEXSPACE ------- 15  REINDEXSPACE KB ------------ 720  REINDEXLEVELS ---- 2

DATAREFAT --------------------- 31  ALLOCATION
SPACE -------------- 15  SPACE KB ----------------- 720  NUMDATASETS ------------ 1

COLUMN ------------------------- SELLER_ID

Example 2 LOAD REPLACE of a partitioned table space

LOADPLUS for DB2 Reference Manual
Example 3 - Single-phase LOAD RESUME YES of a partitioned table space

This example shows a single-phase LOAD RESUME YES job for the same table space that was loaded in example 2.

LOADPLUS adds new data to two of the ten partitions.

The PRELOAD=LOAD default value in the installation options module tells LOADPLUS to use single-phase processing.

In this example, the SYSREC ddname, work file ddnames, and copy ddnames are all overridden by using the appropriate command options in addition to specifying the override ddnames in the JCL. Note the following override names:

- INDATA overrides SYSREC

This example is single-phase LOAD RESUME YES for the same table space that was loaded in example 2. LOADPLUS adds new data to two of the ten partitions.

The PRELOAD=LOAD default value in the installation options module tells LOADPLUS to use single-phase processing.

In this example, the SYSREC ddname, work file ddnames, and copy ddnames are all overridden by using the appropriate command options in addition to specifying the override ddnames in the JCL. Note the following override names:

- INDATA overrides SYSREC
Example 3 - Single-phase LOAD RESUME YES of a partitioned table space

- DATAWK overrides SORTOUT
- ERRWK overrides SYSERR
- IXWK overrides SYSUT1
- DISCARDS overrides SYSDISC
- CPY overrides BMCCPY
- RCY overrides BMCRCY

Assuming that the data was sorted before this run, the LOAD command includes the ORDER NO option. Both indexes are built independently of the data, which allows LOADPLUS to use two SYSUT1 data sets to process the two indexes concurrently.

DDTYPE LOCPFCPY ACTIVE NO disables dynamic allocation for the primary copy data set, overriding the default in the installation options module. The DD statements in the JCL direct LOADPLUS to create two copies of each partition.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

The following figure shows the JCL for example 3:

```plaintext
//EXAMPLE3 EXEC PGM=AMUUMAIN,
  //     PARM='&SSID,EXAMPLE3,NEW/RESTART,,MSGLEVEL(1),AMU$OPTO'
  //     SSID,UTILID ,RSTART,NOTIFY,MSGLEVEL
//STEPLIB DD DISP=SHR,DSN=product.libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// SYSPRINT DD SYSOUT=*  
// SYSPRINT DD SYSOUT=*  
// SYSOUT DD SYSOUT=*  
// SYSDUMP DD SYSOUT=*  
// INDATA DD DSN=AMU.VQA.EXAMPL03.SYSREC,DISP=SHR  
// DATAWK DD DSN=AMU.EXAMP003.SORTOUT,  
//         UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)  
// ERRWK DD DSN=AMU.EXAMP003.SYSERR,  
//         UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)  
// DISCARDS DD DSN=AMU.EXAMP003.SYSDISC,  
//         UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)  
// IXWK01 DD DSN=AMU.EXAMP003.SYSUT1,  
//         UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)  
// IXWK02 DD DSN=AMU.EXAMP003.SYSUT2,  
//         UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)  
// IXWK03 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))  
// IXWK04 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))  
// IXWK05 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))  
// IXWK06 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))  
// IXWK07 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))  
// IXWK08 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))  
// IXWK09 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))  
// IXWK10 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))  
// IXWK11 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))  
```
Example 3 - Single-phase LOAD RESUME YES of a partitioned table space

Chapter 5  Examples of LOADPLUS jobs  509
Example 3 - Single-phase LOAD RESUME YES of a partitioned table space

### SYSPRINT for example 3

The following figure shows the SYSPRINT output for example 3:
Example 3 - Single-phase LOAD RESUME YES of a partitioned table space
Example 3 - Single-phase LOAD RESUME YES of a partitioned table space

LOADPLUS for DB2 Reference Manual
Example 4 LOAD REPLACE of three tables

This example illustrates a LOAD REPLACE of three tables, all with multiple indexes.

This job loads three tables in a single, segmented table space. Each table has a unique clustering index and a non-unique nonclustering index. LOADPLUS replaces all rows in the table space and deletes and redefines the VSAM data sets.

Because the LOAD command does not specify a value for the UNIQUECHECK option, LOADPLUS defaults to UNIQUECHECK YES, which tells LOADPLUS to check the indexes for uniqueness in the PRELOAD phase.

Because each input record is used to create no more than one row, the LOAD command in this example includes the UNIQUEINTO YES option for faster processing.

The COPY YES option tells LOADPLUS to make two image copies of the table space. By default, LOADPLUS registers the copies in the DB2 catalog.

RULES=BMC is specified in the options module so that the job can use extended operators, such as greater than (>) and less than (<), on the LOAD command.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

The following figure shows the JCL for example 4:

```
//EXAMPLE4 EXEC PGM=AMUUMAIN,
//     PARM='&SSID,EXAMPLE4,NEW/RESTART,,MSGLEVEL(1),AMU$OPTS'
//     *                  SSID,UTILID ,RSTART,NOTIFY,MSGLEVEL
//STEPLIB DD DISP=SHR,DSN=product.libraries
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//SYSREC DD DSN=AMU.VQA.EXAMPL04.SYSREC,DISP=SHR
//SORTOUT DD DSN=AMU.EXAMP004.SORTOUT,
//          UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
```

514  LOADPLUS for DB2 Reference Manual
SYSERR DD DSN=AMU.EXAMP004.SYSERR,
// UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
// SYSDISC DD DSN=AMU.EXAMP004.SYSDISC,
// UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
*
SYSUT101 DD DSN=AMU.EXAMP004.SYSUT1,
// UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
// SYSUT102 DD DSN=AMU.EXAMP004.SYSUT2,
// UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
// SYSUT103 DD DSN=AMU.EXAMP004.SYSUT3,
// UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
*
BMCCPY DD DSN=AMU.EXAMP004.COPY1,
// UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
// BMCCPZ DD DSN=AMU.EXAMP004.COPY2,
// UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
// SORTWK01 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
// SORTWK02 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
// SORTWK03 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
// SORTWK04 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
// SORTWK05 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
// SORTWK06 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
// SORTWK07 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
// SORTWK08 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
// SORTWK09 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
// SORTWK10 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
// SORTWK11 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
// SORTWK12 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
*
SYSPRINT DD SYSOUT=*
// SYSOUT DD SYSOUT=*
// UTPRINT DD SYSOUT=*
// SYSUDUMP DD SYSOUT=*
// SYSIN DD *
LOAD REPLACE ORDER YES
PRELOAD CONTINUE
UNIQUEINTO YES
INTO TABLE EXAMPLE4.RET_EMPLS
WHEN STATUS_IND = 'RETIRED'
AND (EMPL_HIREDT >= '1960-01-01'
 OR EMPL_DOB < '1936-01-01')
(EMPL_ID POSITION(*) INTEGER EXTERNAL(4),
 EMPL_LNAME POSITION(*) CHARACTER(20),
 EMPL_FNAME POSITION(*) CHARACTER(10),
 EMPL_DOB POSITION(*) DATE-2E EXTERNAL,
 EMPL_SALARY POSITION(*) DEC EXTERNAL(10),
 EMPL_HIREDT POSITION(*) DATE-2E EXTERNAL,
 STATUS_IND POSITION(*) CHARACTER(7)
)
INTO TABLE EXAMPLE4.SAL_EMPLS
WHEN STATUS_IND = 'SALARY'
(EMPL_ID POSITION(*) INTEGER EXTERNAL(4),
 EMPL_LNAME POSITION(*) CHARACTER(20),
 EMPL_FNAME POSITION(*) CHARACTER(10),
 EMPL_DOB POSITION(*) DATE-2E EXTERNAL,
 EMPL_SALARY POSITION(*) DEC EXTERNAL(10),
 EMPL_HIREDT POSITION(*) DATE-2E EXTERNAL,
 STATUS_IND POSITION(*) CHARACTER(7)
)
INTO TABLE EXAMPLE4.HRLY_EMPLS
WHEN STATUS_IND = 'HOURLY'
(EMPL_ID POSITION(*) INTEGER EXTERNAL(4),
 EMPL_LNAME POSITION(*) CHARACTER(20),
 EMPL_FNAME POSITION(*) CHARACTER(10),
 EMPL_DOB POSITION(*) DATE-2E EXTERNAL,
 EMPL_SALARY POSITION(*) DEC EXTERNAL(10),
 EMPL_HIREDT POSITION(*) DATE-2E EXTERNAL,
Example 4 LOAD REPLACE of three tables

The following figure shows the SYSPRINT output for example 4:

```
  STATUS_IND    POSITION(*)    CHARACTER(7)
    LOG           NO             COPY           YES
```

```
LOADPLUS for DB2  V11.02.00

LOG NO
COPY YES

SYSPRINT for example 4

Example 4 LOAD REPLACE of three tables

The following figure shows the SYSPRINT output for example 4:
Chapter 5  Examples of LOADPLUS jobs  517

Example 4 LOAD REPLACE of three tables
Example 4 LOAD REPLACE of three tables

LOADPLUS for DB2 Reference Manual

518
Example 4 LOAD REPLACE of three tables
Example 5 - LOAD REPLACE with data conversion

This example illustrates a LOAD REPLACE of a partitioned table space with four partitions, a clustering index, and a nonpartitioned index. LOADPLUS replaces all existing rows.

The ORDER YES option tells LOADPLUS to sort the input data in clustering index order. Although LOADPLUS makes four copies of the table space, only two of the copies are registered as specified by the REGISTER option. INLINE NO tells LOADPLUS to make standard image copies instead of inline copies, overriding the value in the installation options module.

Options on the WHEN statement direct LOADPLUS to trim excess space on VARCHAR columns, and convert dates in MM/DD/YY format to the DB2 date format.

This example shows use of a conversion exit that converts a four-byte alphanumeric DEPT_NO to a four-byte numeric field. The STEPLIB DD statement includes the exit library. Member AMUXDEPT of the HLQ.LLQSAMP data set contains this sample exit.

This job also illustrates how you can use LOADPLUS to handle problems with invalid input data. It shows use of the NULLIF clause to bypass the format check in the case of a date input field and a character field. If the HIRE_DATE field contains data that is not in valid date format (***** or 00/00/00), the record is not discarded, but the invalid data is translated to nulls. Likewise, using the NOT IN operand with the PART_401K field tells LOADPLUS to convert invalid data in that field to nulls. The DEFAULTIF clause tells LOADPLUS to convert input values in the PART_401K field to user-assigned default values.

RULES=BMC is specified in the options module so that you can use extended operators, such as greater than (>) and less than (<), on the LOAD command.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

This example provides an image of the DDL that was used to create the DB2 objects being loaded, the JCL that was used to run the job, an image of the SYSREC data set, SYSPRINT output from the execution of the LOADPLUS utility, and output showing the loaded table.
The following figure shows the DDL used to create the DB2 objects being loaded in example 5:

```
CREATE TABLESPACE EX5TS IN AMUEX5DB
  CCSID EBCDIC
  BUFFERPOOL BP0
  CLOSE NO
  NUMPARTS 4
  USING STOGROUP AMUQASTO
    PRIQTY 12
    SECQTY 12
    ERASE NO
  FREEPAGE 005
  PCTFREE 25;
  COMMIT

CREATE TABLE EXAMPLE5.EMPLS
  (EMP_ID             DECIMAL(5) NOT NULL
   ,SSN                 CHAR(9)    NOT NULL
   ,HIRE_DATE           DATE
   ,DEPT_NO             CHAR(4)    NOT NULL
   ,JOB_CODE            SMALLINT
   ,SALARY              DECIMAL(9,2)
   ,PART_401K           CHAR(1)
   ,EMP_LNAME           VARCHAR(20) NOT NULL
   ,EMP_FNAME           VARCHAR(15) NOT NULL
   ,EMP_MNAME           VARCHAR(15)
  )
  IN AMUEX5DB.EX5TS
  PARTITION BY (DEPT_NO)
    (PART 1 VALUES ('3999')
    ,PART 2 VALUES ('5999')
    ,PART 3 VALUES ('7999')
    ,PART 4 VALUES ('9999'));
  COMMIT

CREATE INDEX EXAMPLE5.EMPLS_CLUST_IX
  ON EXAMPLE5.EMPLS (DEPT_NO)
  USING STOGROUP AMUQASTO
  CLUSTER;
  COMMIT

CREATE INDEX EXAMPLE5.EMPLS_NC_IX
  ON EXAMPLE5.EMPLS (EMP_ID)
  PARTITIONED
  USING STOGROUP AMUQASTO;
```

The following figure shows the JCL for example 5:

```
//EXAMPLE5 EXEC PGM=AMUUMAIN,
  //         PARM='&SSID,EXAMPLE5,NEW/RESTART,,MSGLEVEL(1),AMU$OPTS'
  //
  //STEPLIB DD DISP=SHR,DSN=product.libraries
  //         DD DISP=SHR,DSN=AUS.EDITPR.LOADLIB
  //         DD DISP=SHR,DSN=DB2.DSNEXIT
  //         DD DISP=SHR,DSN=DB2.DSNLOAD
  //
  //SYSPRINT DD SYSOUT=* 
  //SYSOUT DD SYSOUT=* 
  //UTPRINT  DD SYSOUT=* 
  //SYSUDUMP  DD SYSOUT=* 
  //
  //SYSREC  DD DSN=AMU.QA.RGRTESTS.DTA163,DISP=SHR
  //SORTOUT  DD DSN=AMU.EXAMPO05.SORTOUT,
             //UNIT=WORK,SPACE=(CYL(1,1)),DISP=(MOD,CATLG,CATLG)
  //SYSERR   DD DSN=AMU.EXAMPO05.SYSERR,
             //UNIT=WORK,SPACE=(CYL(1,1)),DISP=(MOD,CATLG,CATLG)
```
Example 5 - LOAD REPLACE with data conversion

SYSREC for example 5

The following figure shows an image of the SYSREC data set for example 5:

The figure contains a table with several entries, each containing various fields such as EMPLOYEE ID, SSN, JOB CODE, SALARY, DEPARTMENT, and HIRE DATE.
SYSPRINT for example 5:

The following figure shows the SYSPRINT output for example 5:

Example 5 - LOAD REPLACE with data conversion

Chapter 5  Examples of LOADPLUS jobs  523
Example 5 - LOAD REPLACE with data conversion

Chapter 5 Examples of LOADPLUS jobs 525
Example 5 - LOAD REPLACE with data conversion

Example 5 - LOAD REPLACE with data conversion

LOADPLUS for DB2 Reference Manual

LOADPLUS for DB2 Reference Manual
Example 6 LOAD REPLACE with multiple INTO statements

This example illustrates a LOAD REPLACE of a segmented table space that contains one table and one unique, nonclustering index. LOADPLUS replaces all existing rows and deletes and redefines the VSAM data set for the space.

This example uses multiple INTO statements to create a large test table from a small input file and to override input data values. The input data set contains 213,000 records and loaded table from this job contains 852,000 rows.

This LOADPLUS job creates a unique index key by assigning a unique value in KEYFLD1 in each INTO statement. KEYFLD2 is unique for each record of the input file and has values 1 through 213,000.
This example illustrates methods for putting specific values in columns as needed for testing. In this example, the DEFAULTIF option is used to create multiple values in the same column. Different VALUE options are used for the same column in different INTO statements to create multiple values for the same column.

The UNIQUEINTO option is not specified in this job, causing LOADPLUS to use the default, UNIQUEINTO NO. Specifying UNIQUEINTO YES with this job would cause LOADPLUS to stop processing each record after the first INTO statement, thus creating only 213,000 rows instead of 852,000.

The EXEC statement does not supply an installation options module, so LOADPLUS uses the default module, AMUSOPTS. RULES=BMC is specified in the options module so that the job can use extended operators, such as greater than (>) and less than (<), on the LOAD command.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

Because the LOAD command does not include the COPY YES option, LOADPLUS does not make a copy. Because the COPYPEND installation option is set to YES, LOADPLUS places the table space in COPY pending status and the job completes with return code 4.

The following figure shows the JCL for example 6:

```plaintext
//EXAMPLE6 EXEC PGM=AMUUMAIN,
//              PARM='SSID,EXAMPLE6,NEW ,,MSGLEVEL(1)
///                SSID,UTILID ,RSTART,NOTIFY,MSGLEVEL
//STEPLIB DD DISP=SHR,DSN=product.libraries
/// DD DISP=SHR,DSN=DB2.DSNEXIT
/// DD DISP=SHR,DSN=DB2.DSNLOAD
///
///SYSPRINT DD SYSOUT=*
///SYSOUT DD SYSOUT=*  
///UTPRINT DD SYSOUT=* 
///SYSDUMP DD SYSOUT=* 
///SORTOUT DD DSN=AMU.EXAMP006.SORTOUT, 
///          DISP=(MOD,CATLG),UNIT=CART, 
///          VOL=(...10)
///SYSUT101 DD DSN=AMU.EXAMP006.SYSUT1, 
///          UNIT=WORK,SPACE=(CYL,(10,10)),DISP=(MOD,CATLG)
///SORTWK01 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SORTWK02 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SORTWK03 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SORTWK04 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SORTWK05 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SORTWK06 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SORTWK07 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SORTWK08 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SORTWK09 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SORTWK10 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SORTWK11 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SORTWK12 DD UNIT=WORK,SPACE=(CYL,(25,10)),DISP=NEW
///SYSREC DD DSN=AMU.QA.RGRTESTS.DTA111,DISP=SHR
///SYSERR DD DSN=AMU.EXAMP006.SYSERR,
```
Example 6 LOAD REPLACE with multiple INTO statements

LOADPLUS for DB2 Reference Manual
Example 6 LOAD REPLACE with multiple INTO statements

Example 6 LOAD REPLACE with multiple INTO statements
Example 6 LOAD REPLACE with multiple INTO statements

LOADPLUS for DB2 Reference Manual

532
Example 7 LOAD RESUME NO using DISCARDS options and user-defined return codes

In this example, LOADPLUS loads data into a segmented table space that contains only one table with one unique index.

BMCSORT dynamically allocates sort work files. The JCL does not specify any SORTWK DD statements and LOADPLUS dynamic allocation for sort work data sets is not active. The 3390 device type will be used for any sort work files that BMCSORT dynamically allocates (as specified by SORTDEVT 3390).

This example uses the DISCARDS IGNORE option to ignore all duplicate key errors for processing and to keep LOADPLUS from writing errors to the SYSDISC data set. The LOAD command includes the DISCARDS REPORT NO WHEN option. Using this option reduces the number of lines in the SYSPRINT because WHEN discards are not reported in the LOADPLUS error summary report. Not reporting the discarded WHEN records is beneficial if the input data set (SYSREC) contains large amounts of data that will be discarded. Note messages BMC51462I and BMC51463I regarding the DISCARDS IGNORE and REPORT options.

The IFDISCARDS option provides a user-defined return code that LOADPLUS returns if there are any non-ignored discards for the job. The value of 5 overrides the value of 0 for the DISCARDRC installation option.

RESUME NO indicates that the table space being loaded is empty.

SKIPFIELDS YES enables processing of input record fields that do not have corresponding columns in the table. In this case, NULL1 is used in NULLIF condition processing.
Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

The COPY NO command option instructs LOADPLUS not to make a copy. Because the COPYPEND installation option is set to YES, LOADPLUS places the table space in COPY pending status. However, the job completes with return code 5 (instead of the normal return code 4 when COPY pending status is set) because IFDISCARDS 5 is specified and there are discards on the job.

The following figure shows the JCL for example 7:

```
//EXAMPLE7 EXEC PGM=AMUUMAIN,  
// Parm='&SSID,EXAMPLE7,NEW    ,,MSGLEVEL(1),AMU$OPTO'  
// *  
// SYSLIB  DD DISP=SHR,DSN=product.Libraries  
// DD DISP=SHR,DSN=DB2.DSNEXIT  
// DD DISP=SHR,DSN=DB2.DSNLOAD  
// SYSPRINT DD SYSOUT=*  
// SYSPRINT DD SYSOUT=*  
// SYSDUMP DD SYSOUT=*  
// SYSREC DD DSN=AMU.VQA.EXAMPL07.SYSREC,  
// DISP=SHR  
// SORTOUT DD DSN=AMU.EXAMP007.SORTOUT,  
// SPACE=(CYL,(20,10)),UNIT=WORK,DISP=(MOD,CATLG,DELETE)  
// Sysut101 DD DSN=AMU.EXAMP007.SYSUT1,  
// SPACE=(CYL,(20,10)),UNIT=WORK,DISP=(MOD,CATLG,DELETE)  
// SYSERR DD DSN=AMU.EXAMP007.SYSERR,  
// UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,DELETE)  
// SYSDISC DD DSN=AMU.EXAMP007.SYSDISC,  
// UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,DELETE)  
// SYSIN DD *  
LOAD  
RESUME NO  
PRELOAD CONTINUE  
DISCARDS 50 IGNORE DUPKEY REPORT NO WHEN IFDISCARDS 5  
SORTDEVT 3390  
COPY NO  
SKIPFIELDS YES  
INTO TABLE EXAMPLE7.TBL1  
WHEN RECNO <> '90000'  
(RECNO  POSITION(*)  CHARACTER  
,INPUT_A  POSITION(*)  CHARACTER  
,NULL1  POSITION(9)  CHARACTER(1)  
,INPUT_3  POSITION(10:12)  character  
  NULLIF NULL1='?'  
,INPUT_7  POSITION(13:19)  CHARACTER  
,INPUT_10  POSITION(30:39)  CHARACTER  
,INPUT_12  POSITION(25:36)  CHARACTER  
,MLT_SMINT1  POSITION(10)  INTEGER EXTERNAL (3,-1)  
,DIV_SMINT1  POSITION(10)  INTEGER EXTERNAL (3,1)  
,MLT_INT2  POSITION(13)  DECIMAL EXTERNAL (7,-2)  
,DIV_INT2  POSITION(13)  DECIMAL EXTERNAL (7,2)  
,MLT_DEC3  POSITION(25)  DECIMAL EXTERNAL (12,-3)  
,DIV_DEC3  POSITION(25)  DECIMAL EXTERNAL (12,3)  
,MLT_FLOAT40  POSITION(30)  FLOAT EXTERNAL (10,-40)  
,DIV_FLOAT40  POSITION(30)  FLOAT EXTERNAL (10,40)  
)
```
SYSPRINT for example 7

The following figure shows the SYSPRINT output for example 7:

```

Example 7 LOAD RESUME NO using DISCARDS options and user-defined return codes

Chapter 5 Examples of LOADPLUS jobs 535

```
Example 7 LOAD RESUME NO using DISCARD options and user-defined return codes
Example 7 LOAD RESUME NO using DISCARDS options and user-defined return codes

Chapter 5  Examples of LOADPLUS jobs  537
Example 8 LOAD REPLACE with ANALYZE PAUSE

This example shows a LOAD REPLACE of a compressed table space with 10 partitions, a unique clustering index, and a non-unique nonpartitioned index.

The job is run first with ANALYZE PAUSE. You might use this option to obtain the optimal number and size of data sets to use.

Because the installation options module includes AUTOENUMROWS=YES, the command statement does not need to include the ENUMROWS option. Message BMC51561I shows the number of input records that the utility calculated using automated file size estimation.

SYSPRINT A (“Output from the ANALYZE phase (example 8A)” on page 541) shows the ANALYZE report that LOADPLUS generates. The numbers from the ANALYZE report can be used to modify the space allocations on the data set DD statements. The job pauses with return code 4.

The job is run again, with the NEW/RESTART option on the EXEC statement, to perform the actual load. SYSPRINT B (“Output from restarting the job (example 8B)” on page 544) shows this output.

The JCL contains multiple SYSREC and SORTOUT DD statements, enabling LOADPLUS to multitask in both the PRELOAD and LOAD phases. Only the nonpartitioned index is processed by using a SYSUT1 work file, so the JCL includes only one SYSUT1 DD statement.
Because the LOAD command includes the ORDER YES option, LOADPLUS sorts the data rows and partitioned index keys together. Because the LOAD command does not specify a value for the UNIQUECHECK option, LOADPLUS defaults to UNIQUECHECK YES, which tells LOADPLUS to check the clustering index for uniqueness in the PRELOAD phase.

COPY YES INLINE NO REGISTER NONE creates unregistered copies (DSN1COPY-type copies). DDTYPE LOCPFCPY ACTIVE NO disables dynamic allocation for the primary copy data set, overriding the default in the installation options module. The copy DD statements in the JCL direct LOADPLUS to create a single copy of each partition, with each copy in a separate data set.

Specifying CENTURY(1900,1999) overrides the default of (1950,2049) that was specified in the installation option. This override loads the proper century value for records that contain two-digit years for dates earlier than 1950.

This example illustrates how LOADPLUS automatically provides partition parallelism to achieve optimal performance. Message BMC51508I shows that LOADPLUS performs nine data tasks concurrently, with up to two partitions per task.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

LOADPLUS updates statistics in the DB2 catalog. The JCL for this example includes a DD statement for the ASUSRPRT data set. This DD statement tells LOADPLUS not to print the statistics from the Common Statistics component to SYSPRINT, but to send them to a separate data set. “Statistics output (example 8B)” on page 548 shows a portion of this ASUSRPRT data set for 8B.

The following figure shows the JCL for example 8:

```//EXAMPLE8 EXEC PGM=AMUUMAIN, //   PARM='&SSID,EXAMPLE8,NEW        ,,MSGLEVEL(1),AMU$OPTO' //   SSID,UTILID ,RSTART,NOTIFY,MSGLEVEL //STEPLIB DD DISP=SHR,DSN=product.libraries //   DD DISP=SHR,DSN=DB2.DSNEXIT //   DD DISP=SHR,DSN=DB2.DSNLOAD //SYSPRINT DD SYSOUT=* //SYSPUT DD SYSOUT=* //UTPRINT DD SYSOUT=* //SYSUDUMP DD SYSOUT=* //ASUSRPRT DD  DSN=AMU.EXAMP08B.ASUSRPRT, //   UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG) //   DD DISP=SHR,DSN=AMU.QA.RGRTESTS.DTA112 //SYREC2 DD DISP=SHR,DSN=AMU.QA.RGRTESTS.DTA113 //SYREC3 DD DISP=SHR,DSN=AMU.QA.RGRTESTS.DTA114 //SYREC4 DD DISP=SHR,DSN=AMU.QA.RGRTESTS.DTA115 //SORTOUT1 DD DSN=AMU.EXAMP008.SORTOUT1, //   UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)```
Example 8 LOAD REPLACE with ANALYZE PAUSE

//SORTOUT2 DD DSN=AMU.EXAMP008.SORTOUT2,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
//SORTOUT3 DD DSN=AMU.EXAMP008.SORTOUT3,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
//SORTOUT4 DD DSN=AMU.EXAMP008.SORTOUT4,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
//SORTOUT5 DD DSN=AMU.EXAMP008.SORTOUT5,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)

//SYSERR   DD DSN=AMU.EXAMP008.SYSERR,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
//SYSDISC  DD DSN=AMU.EXAMP008.SYSDISC,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)

//SYSUT101 DD DSN=AMU.EXAMP008.SYSUT1,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)

//SORTWK01 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))
//SORTWK02 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))
//SORTWK03 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))
//SORTWK04 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))
//SORTWK05 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))
//SORTWK06 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))
//SORTWK07 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))
//SORTWK08 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))
//SORTWK09 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))
//SORTWK10 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))
//SORTWK11 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))
//SORTWK12 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(5,1))

//BMCCPY01 DD DSN=AMU.EXAMP008.COPY01,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
//BMCCPY02 DD DSN=AMU.EXAMP008.COPY02,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
//BMCCPY03 DD DSN=AMU.EXAMP008.COPY03,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
//BMCCPY04 DD DSN=AMU.EXAMP008.COPY04,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
//BMCCPY05 DD DSN=AMU.EXAMP008.COPY05,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
//BMCCPY06 DD DSN=AMU.EXAMP008.COPY06,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
//BMCCPY07 DD DSN=AMU.EXAMP008.COPY07,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
//BMCCPY08 DD DSN=AMU.EXAMP008.COPY08,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
//BMCCPY09 DD DSN=AMU.EXAMP008.COPY09,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
//BMCCPY10 DD DSN=AMU.EXAMP008.COPY10,
//            UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)

//SYSIN DD *
LOAD REPLACE ORDER YES
PRELOAD CONTINUE
UPDATEDB2STATS YES
CENTURY(1900,1999)
REDEFINE NO
INTO TABLE EXAMPLE8.TBL1
(DATE_OF_SALE POSITION(72:77) DATE-2 EXTERNAL
,SALES_TRANS_ID POSITION(52:61) INTEGER EXTERNAL
,SALES_ITEM_ID POSITION(1:10) CHARACTER
,SALES_QTY POSITION(11:21) INTEGER EXTERNAL
,SALES_PR_PER_ITEM POSITION(22:30) DECIMAL EXTERNAL(9,2)
,SALES_TOTAL_TX POSITION(31:39) DECIMAL EXTERNAL(9,2)
,SALES_TOTAL POSITION(42:50) DECIMAL EXTERNAL(9,2)
,SELLER_ID POSITION(62:71) CHARACTER
)
Example 8 LOAD REPLACE with ANALYZE PAUSE

The following figure shows the SYSPRINT output for example 8:

```
LOG NO  COPY YES INLINE NO REGISTER NONE
ANALYZE PAUSE DDTYPE LOCPFCPY ACTIVE NO
```

### Output from the ANALYZE phase (example 8A)

The Chapter 5 Examples of LOADPLUS jobs section provides an example demonstrating how to use the ANALYZE PAUSE option in a LOADPLUS job. This option is used to analyze data during the load process, which can be particularly useful for identifying and correcting data inconsistencies before the load is completed. The output shown above includes various parameters and options set for the ANALYZE phase, such as COPY, INLINE, and ANALYZE PAUSE. These settings are crucial for ensuring the integrity of the data being loaded into the database.

Incorporating an ANALYZE PAUSE into a LOADPLUS job can help improve the performance and accuracy of the data loading process by allowing the system to analyze the data during key stages of the load. This can be particularly beneficial in scenarios where the data volume is large or the data quality is critical.
Example 8 LOAD REPLACE with ANALYZE PAUSE

LOADPLUS for DB2 Reference Manual
<table>
<thead>
<tr>
<th>BMC51533I SYSDISC</th>
<th>1480</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC51533I SYSERR</td>
<td>630</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>BMC51533I BMCRCZ09</td>
<td>180</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I BMCRCY</td>
<td>1800</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I BMCCPZ</td>
<td>1800</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I BMCCPY</td>
<td>180</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SORTWK</td>
<td>2980</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SYSUT1</td>
<td>426</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SYSUT101</td>
<td>426</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SORTOUT1</td>
<td>234</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SORTOUT3</td>
<td>234</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SORTOUT4</td>
<td>234</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SORTOUT5</td>
<td>234</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SORTOUT6</td>
<td>234</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SORTOUT7</td>
<td>234</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SORTOUT8</td>
<td>234</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SORTOUT9</td>
<td>234</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SORTOUT10</td>
<td>234</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMC51533I SYSUT101</td>
<td>426</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Example 8 LOAD REPLACE with ANALYZE PAUSE

Chapter 5  Examples of LOADPLUS jobs  543
Output from restarting the job (example 8B)

The following figure shows the SYSPRINT output for example 8B:

********** LOAD PLUS FOR DB2 9.223.0.0 **********
(C) COPYRIGHT 1990 - 2016 BMC SOFTWARE, INC.
LOADPLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT NUMBERS 1,664,790 AND 7,930,291

Example 8 LOAD REPLACE with ANALYZE PAUSE

LOADPLUS for DB2 Reference Manual

Example 8 LOAD REPLACE with ANALYZE PAUSE

The following figure shows the SYSPRINT output for example 8B:

********** LOAD PLUS FOR DB2 9.223.0.0 **********
(C) COPYRIGHT 1990 - 2016 BMC SOFTWARE, INC.
LOADPLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT NUMBERS 1,664,790 AND 7,930,291

Example 8 LOAD REPLACE with ANALYZE PAUSE

LOADPLUS for DB2 Reference Manual
Example 8 LOAD REPLACE with ANALYZE PAUSE

Chapter 5  Examples of LOADPLUS jobs  545
Example 8 LOAD REPLACE with ANALYZE PAUSE

Chapter 5  Examples of LOADPLUS jobs  547
Example 8 LOAD REPLACE with ANALYZE PAUSE

The following figure shows a portion of the statistics report sent to ASUSRPRT for example 8B.

This figure shows one set of statistics for each object type. Ellipses (...) at the beginning of a line indicate the omitted statistics sections.

Statistics output (example 8B)
Example 8 LOAD REPLACE with ANALYZE PAUSE

LOADPLUS for DB2 Reference Manual
Example 9 LOAD RESUME YES replacing individual partitions and updating indexes

This example shows a LOAD RESUME YES of the same table that was loaded in example 8.

Data in partition 10 is replaced (PART REPLACE) and additional data is added to partition 9.

This example specifies a separate INTO statement with the PART option for each partition being loaded because only a few partitions are to be updated with new data and RESUME YES is specified. This eliminates building all partitions of the partitioned index and can save considerable time.

Because the LOAD command includes the INDEX UPDATE option, the index entries for the clustering index for partition 9 and the entire nonpartitioned index are updated. The index entries for the clustering index for partition 10 are completely rebuilt, because PART REPLACE is specified for that partition.

The EXEC statement does not supply an installation options module, so LOADPLUS uses the default module, AMU$OPTS.

The SYSREC ddname, work file ddnames, and copy ddnames are all overridden by using the appropriate command options in addition to specifying the override ddnames in the JCL. Note the following override names:

- INDATA overrides SYSREC
- DATAWK overrides SORTOUT
- ERRWK overrides SYSERR
- IXWK overrides SYSUT1
- DISCARDS overrides SYSDISC
- CPY overrides BMCCPY
- RCY overrides BMCRCY

The JCL contains multiple INDATA (SYSREC) and DATAWK (SORTOUT) DD statements, allowing multitasking of the data in both the PRELOAD and LOAD phases.
Assuming that the data was sorted before this run, the LOAD command includes the ORDER NO option. The two indexes are processed independently of the data, which allows LOADPLUS to use two IXWK (SYSUT1) data sets to process the indexes concurrently.

DDTYPE LOCPFCPY ACTIVE NO disables dynamic allocation for the primary copy data set, overriding the default in the installation options module. The COPY YES option and DD statements in the JCL direct LOADPLUS to create two copies of each partition. Note that LOADPLUS makes copies only for the partitions that are participating in the load.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

The following figure shows the JCL for example 9:

```plaintext
//EXAMPLE9 EXEC PGM=AMUUMAIN,
//         PARM='&SSID,EXAMPLE9,NEW/RESTART,,MSGLEVEL(1)'
//*                  SSID,UTILID ,RSTART,NOTIFY,MSGLEVEL
//STEPLIB DD DISP=SHR,DSN=product.libraries
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//         DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD SYSDUMP DD SYSOUT=* 
//SYSPRINT DD SYSDUMP DD SYSOUT=* 
//UTPRINT DD SYSOUT=* 
//SYSUDUMP DD SYSOUT=* 
//*/
//INDATA1 DD DISP=SHR,DSN=AMU.QA.EXAMPLE9.DATA02
//INDATA2 DD DISP=SHR,DSN=AMU.QA.EXAMPLE9.DATA01
//*/
//DATWK1 DD DSN=AMU.EXAMP009.SORTOUT1,
//  UNIT=WORK,SPACE=(CYL,(5,1)),DISP=(MOD,CATLG,CATLG)
//DATWK2 DD DSN=AMU.EXAMP009.SORTOUT2,
//  UNIT=WORK,SPACE=(CYL,(5,1)),DISP=(MOD,CATLG,CATLG)
//ERRWK DD DSN=AMU.EXAMP009.SYSERR,
//  UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
//DISCARDS DD DSN=AMU.EXAMP009.SYSDISC,
//  UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,CATLG,CATLG)
//*/
//IXWK01 DD DSN=AMU.EXAMP009.SYSUT1,
//  UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
//IXWK02 DD DSN=AMU.EXAMP009.SYSUT2,
//  UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,CATLG)
//*/
//SORTWK01 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//SORTWK02 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//SORTWK03 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//SORTWK04 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//SORTWK05 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//SORTWK06 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//SORTWK07 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//SORTWK08 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//SORTWK09 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//SORTWK10 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//SORTWK11 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//SORTWK12 DD UNIT=WORK,DISP=NEW,SPACE=(CYL,(1,1))
//*/
//CPY09 DD DSN=AMU.EXAMP009.COPY09.
```
SYSPRINT for example 9

The following figure shows the SYSPRINT output for example 9:
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value 1</th>
<th>Value 2</th>
<th>Value 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATACLAS</td>
<td>(NONE,NONE)</td>
<td>(NONE,NONE)</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>AVGVOLSP</td>
<td>((30000,TRK),(30000,TRK))</td>
<td>((30000,TRK),(30000,TRK))</td>
<td>((30000,TRK),(30000,TRK))</td>
</tr>
<tr>
<td>SMS</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>DDTYPE</td>
<td>REMPLCPY</td>
<td>REMBLCPY</td>
<td>LOCPXCPY</td>
</tr>
<tr>
<td>GDGSCRAT</td>
<td>N/A</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>GDGEMPTY</td>
<td>N/A</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>GDGLIMIT</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>EXPDT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAXEXTSZ</td>
<td>((0,K),(0,K))</td>
<td>((0,K),(0,K))</td>
<td>((0,K),(0,K))</td>
</tr>
<tr>
<td>THRESHLD</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DSNTYPE</td>
<td>(NONE,NONE)</td>
<td>(NONE,NONE)</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>UNIT</td>
<td>(SYSALLDA,SYSALLDA)</td>
<td>(SYSALLDA,SYSALLDA)</td>
<td>(SYSALLDA,SYSALLDA)</td>
</tr>
<tr>
<td>SIZEPCT</td>
<td>(100,100)</td>
<td>(100,100)</td>
<td>(100,100)</td>
</tr>
<tr>
<td>IFALLOC</td>
<td>USE</td>
<td>USE</td>
<td>USE</td>
</tr>
<tr>
<td>DDTYPE</td>
<td>ERROR</td>
<td>DISCARD</td>
<td>LOCPFCPY</td>
</tr>
<tr>
<td>SMS</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>SIPEXMT</td>
<td>(100,100)</td>
<td>(100,100)</td>
<td>(100,100)</td>
</tr>
<tr>
<td>UNIT</td>
<td>(SYSALLDA,SYSALLDA)</td>
<td>(SYSALLDA,SYSALLDA)</td>
<td>(SYSALLDA,SYSALLDA)</td>
</tr>
<tr>
<td>APCLCOLLECTION</td>
<td></td>
<td>FILECHK=WARN</td>
<td>PRELOAD=LOAD</td>
</tr>
<tr>
<td>EXCLDUMP</td>
<td>(X37,X22,X06)</td>
<td>PLAN=AMUQA</td>
<td>XMLAVGSIZE=10240</td>
</tr>
<tr>
<td>ENFORCE</td>
<td>CHECK</td>
<td>ORIGDISP=DELETE</td>
<td>WORKUNIT=SYSALLDA</td>
</tr>
<tr>
<td>DSNUEXIT</td>
<td>(NONE,ASM)</td>
<td>MSGLEVEL=1</td>
<td>WBUFFS=(20,10)</td>
</tr>
<tr>
<td>DISCDDN</td>
<td>SYSDISC</td>
<td>MAXSORTMEMORY=0</td>
<td>UXSTATE=SUP</td>
</tr>
<tr>
<td>DISCARDRC</td>
<td>0</td>
<td>MAXP=5</td>
<td>TOTALPAGEPCT=30</td>
</tr>
<tr>
<td>DISCARDLIMRC</td>
<td>8</td>
<td>MAPDDN=SYSMAP</td>
<td>TAPEDISP=DELETE</td>
</tr>
<tr>
<td>STOPDELAY</td>
<td>1</td>
<td>LOBAVGPCT=50</td>
<td>SQLDELAY=3</td>
</tr>
<tr>
<td>STOP@CMT</td>
<td>YES</td>
<td>LOCKROW=YES</td>
<td>SQLRETRY=100</td>
</tr>
<tr>
<td>STOPDELAY</td>
<td>1</td>
<td>LOADDN=SORTOUT</td>
<td>CMTRETRY=100</td>
</tr>
<tr>
<td>STOP@CMT</td>
<td>YES</td>
<td>SQLRETRY=100</td>
<td>SQLRETRY=100</td>
</tr>
<tr>
<td>LOADRESUME</td>
<td>YES</td>
<td>CMTRETRY=100</td>
<td>SQLRETRY=100</td>
</tr>
</tbody>
</table>

Example 9 LOAD RESUME YES replacing individual partitions and updating indexes

LOADPLUS for DB2 Reference Manual
Example 9 LOAD RESUME YES replacing individual partitions and updating indexes
Example 10 LOAD RESUME NO of table with check constraints

In this example, LOADPLUS loads data into a partition-by-growth table space with check constraints and one unique index.

Because the value of the ENFORCE installation option is CHECK, LOADPLUS reports check constraint violations. Message BMC51514E displays these violations.

BMCSORT dynamically allocates sort work files. The JCL does not specify any SORTWK DD statements and LOADPLUS dynamic allocation for sort work data sets is not active. The 3390 device type will be used for any sort work files that BMCSORT dynamically allocates (as specified by SORTDEVT 3390).

Example 10 LOAD RESUME NO of table with check constraints
This example uses the DISCARDS IGNORE DUPKEY option to ignore all duplicate key errors for processing and to keep LOADPLUS from writing these errors to the SYSDISC data set. The LOAD command includes the DISCARDS REPORT NO WHEN option. Using this option reduces the number of lines in the SYSPRINT because WHEN discards are not reported in the LOADPLUS error summary report. Not reporting the discarded WHEN records is beneficial if the input data set (SYSREC) contains large amounts of data that will be discarded. Note messages BMC51462I and BMC51463I regarding the DISCARDS IGNORE and REPORT options.

RESUME NO indicates that the table space being loaded is empty. Specifying ENUMROWS 50 overrides the AUTOENUMROWS=YES installation option and estimates that LOADPLUS will load 50 rows.

SKIPFIELDS YES enables processing of input record fields that are not actually in the table. In this case, NULL1 is used in NULLIF condition processing.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

The COPY NO option tells LOADPLUS not to make a copy of the table space. Because the COPYPEND installation option is set to YES, LOADPLUS places the table space in COPY pending status and the job completes with return code 4.

The following figure shows the JCL for example 10:

```jcl
//EXAMPL10 EXEC PGM=AMUUMAIN,  
   //             PARM='&SSID,EXAMPL10,NEW,,MSGLEVEL(1),AMU$OPTS'  
   //                     SSID,UTILID,RSTART,NOSTART,NOTIFY,MSGLEVEL  
//STEPLIB  DD DISP=SHR,DSN=product.libraries  
//         DD DISP=SHR,DSN=DB2.DSNEXIT  
//         DD DISP=SHR,DSN=DB2.DSNLOAD  
//SYSREC   DD DSN=AMU.QA.RGRTESTS(EX10DATA),  
//         DD DISP=SHR  
//SORTOUT  DD DSN=AMU.EXAMP010.SORTOUT,  
//         SPACE=(CYL,(50,10)),UNIT=WORK,DISP=(MOD,CATLG,CATLG)  
//SYSUT101 DD DSN=AMU.EXAMP010.SYSUT1,  
//         SPACE=(CYL,(50,10)),UNIT=WORK,DISP=(MOD,CATLG,CATLG)  
//SYSERR   DD DSN=AMU.EXAMP010.SYSERR,  
//         UNIT=WORK,SPACE=(CYL,(5,1)),DISP=(MOD,CATLG,CATLG)  
//SYSDISC  DD DSN=AMU.EXAMP010.SYSDISC,  
//         UNIT=WORK,SPACE=(CYL,(5,1)),DISP=(NEW,CATLG,CATLG)  
//SYSPRINT DD SYSOUT=*  
//SYSOUT DD SYSOUT=*  
//UTPRINT DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
//SYSIN   DD *  
LOAD  
RESUME NO  
PRELOAD CONTINUE  
DISCARDS 50 IGNORE DUPKEY REPORT NO WHEN  
ENUMROWS 50  
SORTDEVT 3390
```
SYSPRINT for example 10

The following figure shows the SYSPRINT output for example 10:

```
COPY NO SKIPFIELDS YES INTO TABLE EXAMPLEA.TBL1 WHEN RECNO <> '90000'

(RECNO  POSITION(*)  CHARACTER
.INPUT_A  POSITION(*)  CHARACTER
.NULL1    POSITION(9)   CHARACTER(1)
.INPUT_3  POSITION(10:12) CHARACTER

NULIF NULL1=''

.INPUT_7  POSITION(13:19) CHARACTER
.INPUT_10  POSITION(30:39) CHARACTER
.INPUT_12  POSITION(25:36) CHARACTER

.MLT_SMINT1  POSITION(10)  INTEGER EXTERNAL (3,-1)
.DIV_SMINT1  POSITION(10)  INTEGER EXTERNAL (3,1)
.MLT_INT2   POSITION(13)   DECIMAL EXTERNAL (7,-2)
.DIV_INT2   POSITION(13)   DECIMAL EXTERNAL (7,2)
.MLT_DEC3   POSITION(25)   DECIMAL EXTERNAL (12,-3)
.DIV_DEC3   POSITION(25)   DECIMAL EXTERNAL (12,3)
.MLT_FLOAT40 POSITION(30)  FLOAT EXTERNAL (10,-40)
.DIV_FLOAT40 POSITION(30)  FLOAT EXTERNAL (10,40)
```

**Example 10 LOAD RESUME NO of table with check constraints**

Chapter 5  Examples of LOADPLUS jobs  559
Example 11 LOAD REPLACE with dynamic allocation

This example illustrates the use of dynamic allocation during a LOAD REPLACE of a table space with ten partitions, a unique clustering index, and a non-unique nonpartitioned index.
This example loads seven of ten partitions. Individual INTO PART command options are not specified because processing multiple INTO PART command options is less efficient and offers no benefit on a REPLACE run.

The EXEC statement in the JCL includes a parameter to override the installation options module. This options module activates dynamic data set allocation for all copy files by specifying ACTIVE YES for each corresponding DDTYPE. The options module de-activates dynamic allocation for the SYSERR data set by specifying ACTIVE NO for DDTYPE=ERROR.

This example specifies ORDER NO to demonstrate an additional override capability with the IFALLOC option. Because the table has two indexes and ORDER NO is specified, dynamic allocation would normally allocate two SYSUT1 data sets. However, because IFALLOC=USE is specified for SYSUT1 in the options module and the JCL contains a DD statement for a SYSUT101 data set, LOADPLUS uses that data set and dynamically allocates only one additional SYSUT1 data set.

The LOAD statement shows overrides of some of the installation options, including turning on dynamic allocation for the error data set. Also, as a result of specifying IFALLOC=FREE for the error data set, LOADPLUS ignores the DD statement for SYSERR. Because the installation options module includes AUTOENUMROWS=YES, the command statement does not need to include the ENUMROWS option. Message BMC51561I shows the number of input records that the utility calculated using automated file size estimation.

Additional dynamic allocation options perform the following functions:

- **DSNPAT** creates unique data set names.
- **SIZEPCT** tells LOADPLUS to allocate 500 percent of the primary and 50 percent of the secondary data set size calculation.
- **UNIT** tells LOADPLUS what type of device to use for allocation.
- **UNITCNT** tells LOADPLUS how many devices to allocate.
- **VOLCNT** tells LOADPLUS the maximum number of volumes to use during dynamic allocation.
- The copy data sets are allocated with GDG names. The GDGLIMIT option tells LOADPLUS to keep five generations of each copy data set.

This job allocates multiple SORTOUT and SORTWK data sets, demonstrating how LOADPLUS automatically multitasks for optimal performance when you use dynamic allocation.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified.
LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

LOADPLUS updates statistics in both the DB2 catalog and the DASD MANAGER PLUS database statistics tables. The JCL for this example includes the DD statement //ASUSRPRRT DD DUMMY. This DD statement tells LOADPLUS not to produce a report of the statistics.

The following figure shows the JCL for example 11:

```
//EXAMPL11 EXEC PGM=AMUUMAIN,
//          PARM='&SSID,EXAMPL11,NEW        ,,MSGLEVEL(1),AMU$OPTH'
//                  SSID,UTILID ,RSTART,NOTIFY,MSGLEVEL
//STEPLIB DD DISP=SHR,DSN=product.libraries
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//         DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD SYSOUT=* //SYSPRINT DD SYSOUT=* //UTPRINT DD SYSOUT=* //SYSDUMP DD SYSOUT=* //ASUSRPRRT DD DUMMY

//SYSREC DD DSN=AMU.VQA.EXAMPL11.SYSREC,DISP=SHR
//SYSSERR DD DSN=AMU.EXAMP011.SYSSERR,
//         UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,DELETE)
//SYSUT101 DD DSN=AMU.EXAMP011.SYSUT101,
//         UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(MOD,CATLG,DELETE)

//SYSIN DD *
LOAD REPLACE ORDER NO
PRELOAD CONTINUE
UPDATEDB2STATS YES BMCSTATS YES
COPY YES INLINE NO
REDEFINE NO
INTO TABLE EXAMPLEB.TBL1
(DATE_OF_SALE        POSITION(72:77) DATE-2 EXTERNAL,
SALES_TRANS_ID      POSITION(52:61) INTEGER EXTERNAL,
SALES_ITEM_ID       POSITION(1:10)  CHARACTER,
SALES_QTY           POSITION(11:21) INTEGER EXTERNAL,
SALES_PR_PER_ITEM   POSITION(22:30) DECIMAL EXTERNAL(9,2),
SALES_TOTAL_TX      POSITION(31:39) DECIMAL EXTERNAL(9,2),
SALES_TOTAL         POSITION(42:50) DECIMAL EXTERNAL(9,2),
SELLER_ID           POSITION(62:71) CHARACTER)
LOG NO
DDTYPE LOAD     THRESHLD 50000 UNIT(WORK,CART)
VOLCXT 100 UNITCXT 15
DDTYPE SORTWORK DSNPAT 'NONE'
DDTYPE DISCARD DSNPAT 'AMU.&UTIL..T&TIME.&DDNAME'
VOLCXT 100 UNITCXT 15
DDTYPE ERROR    ACTIVE YES IFALLOC FREE SIZEPCT(500,50)
VOLCXT 100 UNITCXT 15
DDTYPE LOCPFCPY
DSNPAT 'AMU.EXAMP011.LOCPFCPY.&DDNAME.(+1)'
DDTYPE LOCBFCPY
DSNPAT 'AMU.EXAMP011.LOCBFCPY.&DDNAME.(+1)'
DDTYPE REMPFCPY
DSNPAT 'AMU.EXAMP011.REMPFCPY.&DDNAME.(+1)'
DDTYPE REMBFCPY
DSNPAT 'AMU.EXAMP011.REMBFCPY.&DDNAME.(+1)'
```
SYSPRINT for example 11

The following figure shows the SYSPRINT output for example 11:
### Example 11 LOAD REPLACE with dynamic allocation

```sql
LOADPLUS for DB2 Reference Manual
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNITCNT</td>
<td>(0,0)</td>
</tr>
<tr>
<td>VOLCNT</td>
<td>(25,25)</td>
</tr>
<tr>
<td>ARVOLSOP</td>
<td>(130000,TRK),(300000,TRK)</td>
</tr>
<tr>
<td>SMUNIT</td>
<td>NO</td>
</tr>
<tr>
<td>SMEUNIT</td>
<td>NO</td>
</tr>
<tr>
<td>SIZEPCT</td>
<td>(100,100)</td>
</tr>
<tr>
<td>UNIT1</td>
<td>(SYSLALDA,SYSLALDA)</td>
</tr>
<tr>
<td>UNITCNT1</td>
<td>(0,0)</td>
</tr>
<tr>
<td>VOLCNT1</td>
<td>(25,25)</td>
</tr>
<tr>
<td>ARVOLSOP1</td>
<td>(130000,TRK),(300000,TRK)</td>
</tr>
<tr>
<td>DTYPE</td>
<td>REMPXCPY</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>DISCARD</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>ERROR</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORK</td>
<td>DSNPAT=&amp;UID..BMC.&amp;ATS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGSCRAT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGEMPTY</td>
<td>NO</td>
</tr>
<tr>
<td>GDGLIMIT</td>
<td>5</td>
</tr>
<tr>
<td>EXPDT</td>
<td>N/A</td>
</tr>
<tr>
<td>MAXEXTSZ</td>
<td>((0,K),(0,K))</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DSNTYPE</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>VOLCNT</td>
<td>(25,25)</td>
</tr>
<tr>
<td>ARVOLSOP</td>
<td>(130000,TRK),(300000,TRK)</td>
</tr>
<tr>
<td>DTYPE</td>
<td>REMPXCPY</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>DISCARD</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>ERROR</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORK</td>
<td>DSNPAT=&amp;UID..BMC.&amp;ATS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGSCRAT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGEMPTY</td>
<td>NO</td>
</tr>
<tr>
<td>GDGLIMIT</td>
<td>5</td>
</tr>
<tr>
<td>EXPDT</td>
<td>N/A</td>
</tr>
<tr>
<td>MAXEXTSZ</td>
<td>((0,K),(0,K))</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DSNTYPE</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>VOLCNT</td>
<td>(25,25)</td>
</tr>
<tr>
<td>ARVOLSOP</td>
<td>(130000,TRK),(300000,TRK)</td>
</tr>
<tr>
<td>DTYPE</td>
<td>REMPXCPY</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>DISCARD</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>ERROR</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORK</td>
<td>DSNPAT=&amp;UID..BMC.&amp;ATS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGSCRAT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGEMPTY</td>
<td>NO</td>
</tr>
<tr>
<td>GDGLIMIT</td>
<td>5</td>
</tr>
<tr>
<td>EXPDT</td>
<td>N/A</td>
</tr>
<tr>
<td>MAXEXTSZ</td>
<td>((0,K),(0,K))</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DSNTYPE</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>VOLCNT</td>
<td>(25,25)</td>
</tr>
<tr>
<td>ARVOLSOP</td>
<td>(130000,TRK),(300000,TRK)</td>
</tr>
<tr>
<td>DTYPE</td>
<td>REMPXCPY</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>DISCARD</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>ERROR</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORK</td>
<td>DSNPAT=&amp;UID..BMC.&amp;ATS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGSCRAT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGEMPTY</td>
<td>NO</td>
</tr>
<tr>
<td>GDGLIMIT</td>
<td>5</td>
</tr>
<tr>
<td>EXPDT</td>
<td>N/A</td>
</tr>
<tr>
<td>MAXEXTSZ</td>
<td>((0,K),(0,K))</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DSNTYPE</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>VOLCNT</td>
<td>(25,25)</td>
</tr>
<tr>
<td>ARVOLSOP</td>
<td>(130000,TRK),(300000,TRK)</td>
</tr>
<tr>
<td>DTYPE</td>
<td>REMPXCPY</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>DISCARD</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>ERROR</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORK</td>
<td>DSNPAT=&amp;UID..BMC.&amp;ATS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGSCRAT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGEMPTY</td>
<td>NO</td>
</tr>
<tr>
<td>GDGLIMIT</td>
<td>5</td>
</tr>
<tr>
<td>EXPDT</td>
<td>N/A</td>
</tr>
<tr>
<td>MAXEXTSZ</td>
<td>((0,K),(0,K))</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DSNTYPE</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>VOLCNT</td>
<td>(25,25)</td>
</tr>
<tr>
<td>ARVOLSOP</td>
<td>(130000,TRK),(300000,TRK)</td>
</tr>
<tr>
<td>DTYPE</td>
<td>REMPXCPY</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>DISCARD</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>ERROR</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORK</td>
<td>DSNPAT=&amp;UID..BMC.&amp;ATS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGSCRAT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGEMPTY</td>
<td>NO</td>
</tr>
<tr>
<td>GDGLIMIT</td>
<td>5</td>
</tr>
<tr>
<td>EXPDT</td>
<td>N/A</td>
</tr>
<tr>
<td>MAXEXTSZ</td>
<td>((0,K),(0,K))</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>DSNTYPE</td>
<td>(NONE,NONE)</td>
</tr>
<tr>
<td>VOLCNT</td>
<td>(25,25)</td>
</tr>
<tr>
<td>ARVOLSOP</td>
<td>(130000,TRK),(300000,TRK)</td>
</tr>
<tr>
<td>DTYPE</td>
<td>REMPXCPY</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>ATTIVENAME</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>DISCARD</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>ERROR</td>
<td>DSNPAT=&amp;UID..BMC.&amp;TS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORK</td>
<td>DSNPAT=&amp;UID..BMC.&amp;ATS..&amp;DDNAME</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>WORKEVENT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGSCRAT</td>
<td>NO</td>
</tr>
<tr>
<td>GDGEMPTY</td>
<td>NO</td>
</tr>
<tr>
<td>GDGLIMIT</td>
<td>5</td>
</tr>
<tr>
<td>EXPDT</td>
<td>N/A</td>
</tr>
<tr>
<td>MAXEXTSZ</td>
<td>((0,K),(0,K))</td>
</tr>
</tbody>
</table>
```
Example 11 LOAD REPLACE with dynamic allocation

Chapter 5 Examples of LOADPLUS jobs  567
Example 11 LOAD REPLACE with dynamic allocation
Example 12 LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY of a partitioned table space

In this example, LOADPLUS uses one of the online load options to load data into a table space that has five partitions.

The LOAD command statement includes the SHRLEVEL CHANGE SQLAPPLY option, which overrides the SHRLEVEL=(NONE,NONE) installation option value.

SYSPRINT A ("Output from ORDER NO (example 12A)" on page 571) shows the output when this job is run with ORDER NO. Because ORDER NO is specified, LOADPLUS sorts only error information for this type of load.

For the second part of this example, the LOAD command is changed to specify ORDER YES. In addition, the EXEC statement in the JCL includes an options module...
that activates dynamic data set allocation for all data sets by specifying ACTIVE YES for each DDTYPE.

The LOAD command provides the following overrides for dynamic allocation:

- Specifying ENUMROWS 100000 overrides the AUTOENUMROWS=YES installation option and estimates that LOADPLUS will load 100000 rows.
- The DSNPAT command option overrides the pattern for the work file data set names.
- The DELETEFILES YES SYSDISC YES option tells LOADPLUS to delete the work files and SYSDISC file after the load completes.

SYSPRINT B (“Output from ORDER YES with dynamic allocation (example 12B)” on page 577) shows the output from the ORDER YES job.

In both jobs, message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

Note the BMC51576 messages, which return information from the High-speed Apply Engine. The High-speed Apply Engine message numbers (for example, APT0166I) are displayed to the right of the BMC51576 message number and timestamp. For message explanations, access the BMC Documentation Center from the BMC Support Central site (http://www.bmc.com/support).

In both jobs, High-speed Apply Engine message APT0166I displays a report of the number of SQL statements (for LOADPLUS, these will always be INSERT statements) committed per agent. Because the value of the APMULTIROW installation option is 100, each INSERT statement contains 100 input rows. Therefore, the number of statements committed is the number of input rows divided by 100.

Although the value of the COPIPEND default installation option is YES, LOADPLUS does not set COPY pending because COPIPEND YES is ignored for an SQLAPPLY load job.

The following figure shows the JCL for example 12A:
The following figure shows the JCL for example 12B:

```
//EXAMP12B EXEC PGM=AMUUMAIN,
//         PARM='&SSID,EXAM12B,NEW    ,,MSGLEVEL(1),AMU$OPTE'
//*                  SSID,UTILID ,RSTART,NOTIFY,MSGLEVEL
//STEPLIB  DD DISP=SHR,DSN=product.libraries
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//         DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD SYSOUT=* 
//SYSOUT   DD SYSOUT=* 
//UTPRINT  DD SYSOUT=* 
//APTDUMP  DD SYSOUT=* 
//SYSUDUMP DD SYSOUT=* 
//SYSREC   DD  DSN=AMU.QA.RGRTESTS.DTAEX12,DISP=SHR
//SYSIN    DD   * 
LOAD DATA RESUME YES 
SHRLEVEL CHANGE 
ENUMROWS(100000) 
ORDER YES 
DDTYPE LOAD     DSNPAT '&UID..RESUME&RESUME.&DDNAME'
DDTYPE WORK     DSNPAT '&UID..RESUME&RESUME.&DDNAME'
DDTYPE SORTWORK DSNPAT '&UID..RESUME&RESUME.&DDNAME'
DDTYPE DISCARD  DSNPAT '&UID..RESUME&RESUME.&DDNAME'
DDTYPE ERROR    DSNPAT '&UID..RESUME&RESUME.&DDNAME'
DELETEFILES YES SYSDISC YES 
INTO TABLE EXAMPLEC.TBL1 
(RECNO      POS(1:4)   INTEGER,
  INPUT_A    POS(5:14)  CHAR(10),
  RECNO2     POS(15:18) INTEGER,
  INPUT_B    POS(19:28) CHAR(10),
  INPUT_C    POS(29:43) CHAR(15),
  INPUT_D    POS(44:58) CHAR(15)
)
```

The following figure shows the SYSPRINT output for example 12A:

```
1                          *************  B M C   L O A D P L U S   F O R   D B 2    V11R2.00 ************
(C) COPYRIGHT 1990 - 2015 BMC SOFTWARE, INC.
LOADPLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT NUMBERS 7,664,790 AND 7,930,291
-BMC50001I UTILITY EXECUTION STARTING    3/09/2015   15:00:58 ...
-BMC50002I UTILITY ID = 'EXAM12A'.  DB2 SUBSYSTEM ID = 'DEHJ'.  OPTION MODULE = 'AMU$OPTO'.
-BMC50471I z/OS 2.1.0,PID=HBB7790,DFSMS FOR Z/OS=2.1.0,DB2=11.1.0
-BMC50471I REGION=0M,BELOW 16M=8816K,ABOVE 16M=1306040K,IEFUSI=NO,CPUS=3
-BMC50471I MEMLIMIT=17592186040320M,AVAILABLE=17592186040308M,MEMLIMIT SET BY:REGION=0
-BMC50471I LOADPLUS FOR DB2--V11.02.00
```

Output from ORDER NO (example 12A)

The following figure shows the SYSPRINT output for example 12A:
Example 12 LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY of a partitioned table space

Chapter 5 Examples of LOADPLUS jobs 573
Example 12 LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY of a partitioned table space

LOADPLUS for DB2 Reference Manual
Example 12 LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY of a partitioned table space

Chapter 5  Examples of LOADPLUS jobs  575
Example 12 LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY of a partitioned table space
Output from ORDER YES with dynamic allocation (example 12B)

The following figure shows the SYSPRINT output for example 12B:

Example 12 LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY of a partitioned table space
Example 12 LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY of a partitioned table space

Chapter 5  Examples of LOADPLUS jobs  579
Example 12 LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY of a partitioned table space

Chapter 5 Examples of LOADPLUS jobs 581
<table>
<thead>
<tr>
<th>Command</th>
<th>Action</th>
<th>Owner</th>
<th>Qualifier</th>
<th>Validate</th>
<th>Explain</th>
<th>Isolation</th>
<th>Release</th>
<th>Commit</th>
<th>Prepara</th>
<th>Execute</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOADPLUS</td>
<td>YES</td>
<td>SHRLEVEL</td>
<td>CHANGE</td>
<td>SQLAPPLY</td>
<td>YES</td>
<td>RESUME</td>
<td>YES</td>
<td>SHRLEVEL</td>
<td>CHANGE</td>
<td>SQLAPPLY</td>
</tr>
</tbody>
</table>

Example 12 LOAD RESUME YES SHLEVEL CHANGE SQLAPPLY of a partitioned table space

LOADPLUS for DB2 Reference Manual
Example 13 LOAD REPLACE from CSV input

This example shows a LOAD REPLACE of a segmented table space from CSV input. There is one unique clustering index on the table.

The PRELOAD=LOAD default value in the installation options module tells LOADPLUS to use single-phase processing.

This example provides an image of the SYSREC data set that contains the CSV input. The input file uses the default delimiter (,) and enclosure characters (''), so the FORMAT CSV option does not need any additional parameters. The input file uses a
fixed-length record format, so a delimiter was added to the end of each record to ensure that the load job does not encounter end-of-record errors.

For FORMAT CSV, LOADPLUS uses RULES=STANDARD. In this example, message BMC51414I, tells you that LOADPLUS has changed the value of the RULES installation option to STANDARD.

The COPY NO COPYPEND NO specifications tells LOADPLUS not to make a copy and not to place the table space in COPY pending status following the load. Because the value for the COPYPEND installation option is YES and not (YES,ENFORCE), the COPYPEND NO command option can override the YES value.

The field specification for the FLOAT_COLUMN column tells LOADPLUS to interpret any two consecutive delimiter characters in this column as null.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

This example also provides the SPUFI output that shows the loaded table.

The following figure shows the JCL for example 13:

```
//EXAMPL13 EXEC PGM=AMUUMAIN,
//             PARM='&SSID,AMUEXDDB,NEW,,MSGLEVEL(1),AMU$OPTS'
//STEPLIB DD DISP=SHR,DSN=product.libraries
//       DD DISP=SHR,DSN=DB2.DSNEXIT
//       DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD SYSOUT=*  //SYSOUT DD SYSOUT=*  //UTPRINT DD SYSOUT=* //SYSUDUMP DD SYSOUT=*  //SORTOUT DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(NEW,PASS),  //       DSN=AMU.EXAMP013.SORTOUT //SYST101 DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(NEW,PASS),  //       DSN=AMU.EXAMP013.SYST1  //SYSSERR DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(NEW,PASS),  //       DSN=AMU.EXAMP013.SYSSERR  //SYSDISC DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(NEW,PASS),  //       DSN=AMU.EXAMP013.SYSDISC  //SYSEXEC DD DSN=AMU.QA.RGRTESTS.DTAEX13,DISP=SHR  //SYSIN DD *  //SYSIN DD *  LOAD REPLACE COPY NO COPYPEND NO  //FORMAT CSV INTO TABLE EXAMPL13.TBL1  (INTEGER_COLUMN POSITION(*),  SMALLINT_COLUMN POSITION(*),  DECIMAL_COLUMN POSITION(*),  CHAR_COLUMN POSITION(*),  VARCHAR_COLUMN POSITION(*),  DATE_COLUMN_D1 POSITION(*) DATE-D1 EXTERNAL  ,TIME_COLUMN_D1 POSITION(*) TIME-D1 EXTERNAL  ,TIMESTP_COLUMN_D1E POSITION(*) TIMESTAMP-1E EXTERNAL
```

Example 13 LOAD REPLACE from CSV input
Example 13 LOAD REPLACE from CSV input

The following figure shows the SYSPRINT output for example 13:

```
%RUN (LOAD PLUS, LOAD+) LOAD REPLACE FROM CSV INPUT

Example 13 LOAD REPLACE from CSV input

The following figure shows the SYSPRINT output for example 13:
```

SYSREC for example 13

The following figure shows the SYSREC data set for example 13:

```
Example 13 LOAD REPLACE from CSV input

The following figure shows the SYSREC data set for example 13:
```

SYSPRINT for example 13

The following figure shows the SYSPRINT output for example 13:

```
%RUN (LOAD PLUS, LOAD+) LOAD REPLACE FROM CSV INPUT

Example 13 LOAD REPLACE from CSV input

The following figure shows the SYSPRINT output for example 13:
```

Chapter 5  Examples of LOADPLUS jobs  585
Example 13 LOAD REPLACE from CSV input
SELECT on the table for example 13

The following figure shows the SPUFI output of a SELECT on the loaded table for example 13:

---

Example 14 LOAD REPLACE from UNLOAD PLUS data in internal format

This example is a LOAD REPLACE of a partitioned table space with one non-unique partitioning index that is not a clustering index and one non-unique, nonpartitioned index. No clustering index exists on the table space.

In this example, LOADPLUS loads data that has been unloaded by using UNLOAD PLUS. The UNLOAD PLUS command includes the FORMAT BMCLOAD option to unload the data to an output file in an internal format that only LOADPLUS can read. The LOADPLUS command includes the FORMAT BMCUNLOAD option to
load this data. The combination of these options provides a high-speed data migration solution.

To provide a complete picture of this feature, this example includes the UNLOAD PLUS JCL and SYSPRINT, the LOADPLUS JCL and SYSPRINT, and output showing the loaded table.

The UNLOAD PLUS JCL includes the following options:

- The OUTPUT option provides dynamic allocation options for the default output data set.
- The CNTLCARDS option passes LOADPLUS command options to the SYSCNTL data set. The control cards that are specified with this option will generate a LOAD REPLACE run with dynamic allocation.

**Note**

When you specify FORMAT BMCLOAD in UNLOAD PLUS, UNLOAD PLUS forces CNTLCARDS BMCLOAD.

- The ENUMROWS control card tells UNLOAD PLUS to determine the value to include with the LOADPLUS ENUMROWS option from the number of rows that are unloaded.
- ORDER YES tells UNLOAD PLUS to sort output records in clustering or partitioning key sequence. The full data migration process obtains optimized sort processing by sorting in UNLOAD PLUS rather than in LOADPLUS.
- The INTO parameter of the SELECT statement passes to LOADPLUS the name of the table into which LOADPLUS is to load the data. This table is identical in structure to the table from which UNLOAD PLUS is unloading data.

The UNLOAD PLUS SYSPRINT shows the LOADPLUS control cards that UNLOAD PLUS writes to the SYSCNTL data set. Note the following LOADPLUS options in these control cards:

- The INDSN option names the SYSREC data set that UNLOAD PLUS created. UNLOAD PLUS builds the data set name dynamically based on the DSNAME values in the UNLOAD PLUS installation options.
- The ENUMROWS option shows that UNLOAD PLUS calculated the number of new rows that LOADPLUS will load as 29, and overrides the AUTOENUMROWS=YES installation option. This number includes the header records that UNLOAD PLUS creates when using FORMAT BMCLOAD. LOADPLUS uses these header records to ensure that the table being loaded matches the table that was unloaded.
- DELETEFILES YES SYSDISC YES tells LOADPLUS to delete the work files and SYSDISC file (if the SYSDISC file is empty) after the load completes.

- The INTO statement contains the WHEN TABLE=obid option, which is required when using FORMAT BMCUNLOAD. The OBID is the identifier that UNLOAD PLUS generated for this table.

- Because no COPY option is included, LOADPLUS does not make a copy. Because the value of the COPYPEND installation option is YES, LOADPLUS places the table space in COPY pending status and the LOADPLUS job completes with return code 4.

The SYSIN DD statement in the LOADPLUS JCL specifies the name of the SYSCNTL data set that contains the LOAD command options that UNLOAD PLUS created. Passing the options from UNLOAD PLUS to LOADPLUS in this way streamlines the data migration process because it eliminates the need to specify options in your LOADPLUS JCL. This method maximizes data integrity by ensuring that the LOADPLUS control cards match the input data. Note that the JCL does not include a SYSREC DD statement because the input data set is specified with the INDSN option.

In the LOADPLUS SYSPRINT, note message BMC50138I that indicates that ENFORCE CHECK CONSTRAINTS (which is set as the default by the ENFORCE installation option) is not allowed when you specify FORMAT BMCUNLOAD. LOADPLUS sets the ENFORCE option to NO in this case.

For both jobs, message 50041I indicates the status of zIIP processing. For this example, the zIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

The following figure shows the UNLOAD PLUS JCL for example 14:

```sql
//UNLOAD14 EXEC PGM=ADUUMAIN,COND=EVEN,
  //       PARM=’&SSID,ADUXM14,NEW,,MSGLEVEL(1)’
//STEPLIB DD DISP=SHR,DSN=product.libraries
//      DD DISP=SHR,DSN=DB2.DSNEXIT
//      DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD SYSOUT=* 
//UTPRINT DD SYSOUT=* 
//SYSCNTL DD DSN=AMU.EXAMP014.SYSCNTL,
  //      DISP=(NEW,CATLG),SPACE=(CYL,(5,2)),
  //      UNIT=SYSDA
//SYSSN DD *
UNLOAD
  OUTPUT SYSREC UNIT SYSDA
  SPACE (5,1)
  DSNAME ‘AMU.EXAMP014.SYSREC’
  CNTLVCARDS ’ENUMROWS’ ’REPLACE’
  ’DELETEFILES YES SYSDISC YES’
  ORDER YES
  FORMAT BMCLOAD
```
Chapter 5  Examples of LOADPLUS jobs  591

SELECT * INTO NAME BMC.TBEXB20
FROM BMC.RET_EMPLS

The following figure shows the LOADPLUS JCL for example 14:

```
//EXAMPL14 EXEC PGM=AMUUMAIN,
// PARM='&SSID,EXAMPLE14,NEW,,MSGLEVEL(1)'
//STEP1B DD DISP=SHR,DSN=product_libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD SYSPUT=* 
//SYSOUT DD SYSPUT=
//UTPRINT DD SYSPUT=
//SYSDUMP DD SYSPUT=
//SYSERR DD DSN=AMU.EXAMPLE14.SYSERR, 
// DISP=(NEW,CATLG).SPACE=(CYL,(5,2)).
// UNIT=SYSDA
//SYSG DD DISP=SHR,DSN=AMU.EXAMP014.SYSCNTL
```
Example 14 LOAD REPLACE from UNLOAD PLUS data in internal format

LOADPLUS SYSPRINT for example 14

The following figure shows the LOADPLUS SYSPRINT output for example 14:

LOADPLUS for DB2 Reference Manual

592
SELECT on the table for example 14

The following figure shows the output of a SELECT on the loaded table for example 14:

---

**Table 1**

<table>
<thead>
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
<th>Column 4</th>
<th>Column 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data 1</td>
<td>Data 2</td>
<td>Data 3</td>
<td>Data 4</td>
<td>Data 5</td>
</tr>
<tr>
<td>Data 1</td>
<td>Data 2</td>
<td>Data 3</td>
<td>Data 4</td>
<td>Data 5</td>
</tr>
<tr>
<td>Data 1</td>
<td>Data 2</td>
<td>Data 3</td>
<td>Data 4</td>
<td>Data 5</td>
</tr>
<tr>
<td>Data 1</td>
<td>Data 2</td>
<td>Data 3</td>
<td>Data 4</td>
<td>Data 5</td>
</tr>
</tbody>
</table>

---

**Table 2**

<table>
<thead>
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
<th>Column 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data 1</td>
<td>Data 2</td>
<td>Data 3</td>
<td>Data 4</td>
</tr>
<tr>
<td>Data 1</td>
<td>Data 2</td>
<td>Data 3</td>
<td>Data 4</td>
</tr>
<tr>
<td>Data 1</td>
<td>Data 2</td>
<td>Data 3</td>
<td>Data 4</td>
</tr>
<tr>
<td>Data 1</td>
<td>Data 2</td>
<td>Data 3</td>
<td>Data 4</td>
</tr>
<tr>
<td>Data 1</td>
<td>Data 2</td>
<td>Data 3</td>
<td>Data 4</td>
</tr>
</tbody>
</table>

---

**Example 14 LOAD REPLACE from UNLOAD PLUS data in internal format**

---

Chapter 5 Examples of LOADPLUS jobs 595
Example 15 LOAD REPLACE ignoring indexes

This example is a LOAD REPLACE of a table space with 10 partitions, a unique clustering index, and both a unique and a non-unique nonpartitioned index.

In this example, LOADPLUS ignores both of the nonpartitioned indexes, which means that LOADPLUS does not check for uniqueness in the unique nonpartitioned index and does not build or update either nonpartitioned index. Message BMC51493W indicates that the two indexes were skipped and placed in REBUILD pending (RBDP) status.

The WHEN option illustrates using a predicate that compares a \texttt{(start:end)} position in the file to a hexadecimal constant.

Because the LOAD command includes the ORDER YES option, LOADPLUS sorts the data rows and clustering index keys. Because the LOAD command does not specify a value for the UNIQUECHECK option, LOADPLUS defaults to UNIQUECHECK YES, which tells LOADPLUS to check the indexes for uniqueness in the PRELOAD phase.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

The COPY NO option tells LOADPLUS not to make a copy of the table space. Because the COPYPEND installation option is set to YES, LOADPLUS places the table space in COPY pending status.

Because the nonpartitioned indexes are placed in RBDP status and the table space is placed in COPY pending status, the job completes with return code 4.

The following figure shows the JCL for example 15:

```bash
//EXAMPL15 EXEC PGM=AMUUMAIN,
//                  PARM='&SSID,EXAMPL15,NEW/RESTART,,MSGLEVEL(1),AMU$OPTO'
//                  SSID,UTILID ,RSTART,NOTIFY,MSGLEVEL
```
Example 15 LOAD REPLACE ignoring indexes

The following figure shows the SYSPRINT output for example 15:
<table>
<thead>
<tr>
<th>BMC50470I</th>
<th>COPYUNE=FULL</th>
<th>LOADUNDSORT=PATH</th>
<th>SALLRET=100</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC50470I</td>
<td>COPYPEND=YES</td>
<td>LOADCPT=50</td>
<td>STOPCMT=YES</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>COPYPEND=NO</td>
<td>LOADCPT=100</td>
<td>STOPCMT=300</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>DEFILEL=NO</td>
<td>USERAM=50,5600</td>
<td>STOPCMT=300</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>DISCARD=NCME=H</td>
<td>MAPDUMP=DELETE</td>
<td>TAPEDUMP=DELETE</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>DISCARD=O</td>
<td>MAPDUMP=DELETE</td>
<td>TOTALPAGECTRL=30</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>DISCARD=SYSDISC</td>
<td>MARGGROUP=0</td>
<td>TSSAMPLECTRL=100</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>DRODELAY=55</td>
<td>MXTEND=CONTINUE</td>
<td>USTADV=0.0</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>DROWAIT=NONE</td>
<td>NOUNGROUP=0</td>
<td>USTDATE=0.0</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>DONUM=NONE,NODES</td>
<td>MXTEND=CONTINUE</td>
<td>WRITF=0.0,1.0</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>DONUM=NONE,NONE</td>
<td>RUNGROUP=DELETE</td>
<td>WRITF=0.0,1.0</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>ENTEREXIT=1</td>
<td>UPTDATA=AUTHID=USER</td>
<td>WRITF=0.0,1.0</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>PRINTCHECK=NO</td>
<td>UPTDATA=DELETE</td>
<td>WRITF=0.0,1.0</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>ERRDISC=SYSERR</td>
<td>PAUSEDISCARDS=0</td>
<td>WRITF=0.0,1.0</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>EXCLUDFILE=EXCLUDE</td>
<td>PLAN=MUPDA</td>
<td>XCLASSCIZE=10240</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>FILECHK=NO</td>
<td>PREFORMAT=NO</td>
<td>XDEF=NO</td>
</tr>
<tr>
<td>BMC50470I</td>
<td>FILECHK=WARB</td>
<td>PREDLOAD=0,0</td>
<td>ZIP=ENABLED</td>
</tr>
</tbody>
</table>

Example 15 LOAD REPLACE ignoring indexes

LOADPLUS for DB2 Reference Manual
Example 15 LOAD REPLACE ignoring indexes
LOADPLUS for DB2 Reference Manual
Example 15 LOAD REPLACE ignoring indexes

Chapter 5 Examples of LOADPLUS jobs

Example 15 LOAD REPLACE ignoring indexes

Chapter 5 Examples of LOADPLUS jobs

Example 15 LOAD REPLACE ignoring indexes

Chapter 5 Examples of LOADPLUS jobs

Example 15 LOAD REPLACE ignoring indexes

Chapter 5 Examples of LOADPLUS jobs
Example 16 LOAD RESUME YES replacing individual tables

This example illustrates replacing data in only two out of three tables in a segmented table space by specifying LOAD RESUME YES with INTO TABLE `tableName` REPLACE.

The table space has one non-unique clustering index and one unique nonclustering index per table.

The PRELOAD=LOAD default value in the installation options module tells LOADPLUS to use single-phase processing.

Because the LOAD command includes the ORDER YES option, LOADPLUS sorts the data rows and clustering index keys.

COPY YES INLINE NO REGISTER ALL tells LOADPLUS to create a standard image copy of the loaded table space and register it with DB2.

Dynamic data set allocation is enabled for most data sets in the installation options module. Additionally, the JCL activates dynamic allocation for sort work data sets by specifying ACTIVE YES for DDTYPE SORTWORK. Because the installation options module includes AUTOENUMROWS=YES, the command statement does not need to include the ENUMROWS option for dynamic allocation. Message BMC51561I shows the number of input records that the utility calculated using automated file size estimation. The LOAD command also includes the DELETEFILES YES SYSDISC YES option to delete the work files, including SYSDISC if it is empty, after the load completes.

RULES=BMC is specified in the options module so that the job can use extended operators, such as greater than (>) and less than (<), on the LOAD command.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

In addition to the JCL and the SYSPRINT output, this example provides the results of a SELECT COUNT on the table space before and after running the load job. The second SELECT output shows that only tables TBREDE2 and TBREDE3 were replaced, leaving table TBREDE1 intact.

The following figure shows the JCL for example 16:

```
//EXAMPL16 EXEC PGM=AMUUMAIN,
//             PARM='&SSID,AMURS001,NEW    ,,MSGLEVEL(1),AMU$OPTS'
//*                     SSID,UTILID ,RSTART,NOTIFY,MSGLEVEL
//STEPLIB  DD DISP=SHR,DSN=
```
The following figure shows the number of rows in each table in the table space before running this job:
Example 16 LOAD RESUME YES replacing individual tables

**SYSPRINT for example 16 and SELECT on the table space after the load**

The following figure shows the SYSPRINT output for example 16:
Example 16 LOAD RESUME YES replacing individual tables
Example 17 Loading into a Unicode table space from EBCDIC input

This example illustrates running a LOAD REPLACE on a segmented table space that is defined as Unicode. No indexes exist on the table space.

The EBCDIC option tells LOADPLUS that the encoding scheme of the input data is EBCDIC, and the CCSID option tells LOADPLUS which code pages were used to
encode the data. LOADPLUS translates the input EBCDIC data into Unicode data during the load process.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

No COPY option is included on the LOAD command, so LOADPLUS assumes the default of COPY NO and does not make a copy. Because the COPYPEND installation option is set to YES, LOADPLUS places the table space in COPY pending status and the job completes with return code 4.

The following figure shows the DDL used to create the Unicode table space:

```
CREATE DATABASE AMUEX17D
  CCSID UNICODE
  STOGROUP AMUQASTO
  BUFFERPOOL BP0 :
COMMIT :
CREATE TABLESPACE EX17TS IN AMUEX17D
  SEGSIZE 64
  CCSID UNICODE
  USING STOGROUP AMUQASTO
    PRIQTY 50
    SECQTY 10
    ERASE NO
  BUFFERPOOL BP0 :
COMMIT :
CREATE TABLE EXAMPL17.TBL1
  (SMINT_COL1 SMALLINT
   ,TIMESTAMP_COL2 TIMESTAMP
   ,DATE_COL3 DATE
   ,CHAR_COL4 CHAR(7)
   FOR SBCS DATA
  ) IN AMUEX17D.EX17TS :
```

The following figure shows the JCL for example 17:

```
//EXAMPL17 EXEC PGM=AMUUMAIN,
//         PARM='&SSID,EXAMPL17,NEW/RESTART,,MSGLEVEL(1),AMU$OPTO'
//             SSID,UTILID,RSTART,NOTIFY,MSGLEVEL
//STEPLIB DD DISP=SHR,DSN=product.libraries
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//         DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD SYSOUT=* 
//SYSOUT DD SYSOUT=* 
//UTPRINT DD SYSOUT=* 
//SYSDUMP DD SYSOUT=* 
//SYSREC DD DISP=SHR,DSN=AMU.VQA.EXAMP17A.SYSREC 
//SORTOUT DD DSN=AMU.EXAMP17.SORTOUT. 
//SYSSERR DD DSN=AMU.EXAMP17.SYSSERR, 
//SYSDISC DD DSN=AMU.EXAMP17.SYSDISC, 
//SYSIN DD *
LOAD REPLACE
```
SYSPRINT for example 17

The following figure shows the SYSPRINT output for example 17:
Example 17 Loading into a Unicode table space from EBCDIC input

LOADPLUS for DB2 Reference Manual

610
Example 17 Loading into a Unicode table space from EBCDIC input

Chapter 5  Examples of LOADPLUS jobs  611
Example 18 LOAD REPLACE enforcing referential constraints

This example illustrates the use of the ENFORCE keyword to check referential constraints on a single-table, segmented table space.

The table space has one unique clustering index, one unique nonclustering index, and two non-unique, nonclustering indexes.

This example performs a LOAD REPLACE, and uses the ENFORCE CONSTRAINTS command option to override the ENFORCE=CHECK installation option. Although ENFORCE CONSTRAINTS checks both referential constraints and check constraints, this table is defined with only referential constraints.

The XBMID command option specifies the XBM subsystem ID to use when enforcing referential constraints. The SYSPRINT shows the messages that display when using XBM or SUF during referential constraint checking. This SYSPRINT also shows the following messages related to enforcing constraints:

- Messages BMC51525I and BMC51526I display the referential constraint violations.
- Message BMC50270I shows that CHKP status is reset because all constraints were checked.

The PRELOAD=LOAD default value in the installation options module tells LOADPLUS to use single-phase processing.

Because the LOAD command includes the ORDER YES option, LOADPLUS sorts the data rows and clustering index keys.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED and an XBM subsystem ID was specified (for referential constraint checking). LOADPLUS uses the same XBM subsystem ID for zIIP processing.

No COPY option is included on the LOAD command, so LOADPLUS does not make a copy. Because the COPYPEND installation option is set to YES, LOADPLUS places the table space in COPY pending status and the job completes with return code 4.

The following figure shows the DDL to create the objects for this example, illustrating the referential relationship between those objects:

```sql
CREATE DATABASE DBEXMRI
BUFFERPOOL BPO;
COMMIT;

CREATE TABLESPACE TSEXMRI IN DBEXMRI
BUFFERPOOL BPO
CLOSE NO
```
CREATE TABLE AMU.EXMTBLA (
  COL1_A         INTEGER NOT NULL WITH DEFAULT CHECK (COL1_A > 10),
  COL2_A         INTEGER NOT NULL,
  COL3_A         CHAR(10) NOT NULL,
  COL4_A         CHAR(10) NOT NULL,
  COL5_A         VARCHAR(300) NOT NULL,
  COL6_A         VARCHAR(300) NOT NULL,
  PRIMARY KEY (COL1_A,COL5_A,COL3_A)
) IN DBEXMRI.TSEXMRI;

CREATE UNIQUE INDEX AMU.IXEXMA1
  ON AMU.EXMTBLA (COL1_A ASC,COL5_A DESC,COL3_A ASC)
  NOT PADDED CLUSTER USING STOGROUP AMUQASTO
  PRIQTY 10000 SECQTY 1000 FREEPAGE 1 PCTFREE 010;

CREATE UNIQUE INDEX AMU.IXEXMA2
  ON AMU.EXMTBLA (COL3_A)
  USING STOGROUP AMUQASTO
  PRIQTY 10000 SECQTY 1000 FREEPAGE 4 PCTFREE 010;

CREATE INDEX AMU.IXEXMA3
  ON AMU.EXMTBLA (COL2_A)
  USING STOGROUP AMUQASTO
  PRIQTY 10000 SECQTY 1000 FREEPAGE 4 PCTFREE 010;

CREATE INDEX AMU.IXEXMA4
  ON AMU.EXMTBLA (COL5_A)
  USING STOGROUP AMUQASTO
  PRIQTY 10000 SECQTY 1000 FREEPAGE 4 PCTFREE 010;

CREATE DATABASE DBEXMRI2 BUFFERPOOL BP0;

CREATE TABLESPACE TSEXMRI2 IN DBEXMRI2
  BUFFERPOOL BP0 CLOSE NO
  SEGSIZE 32 USING STOGROUP AMUQASTO
  PRIQTY 10000 SECQTY 1000
The following figure shows the JCL for example 18:

```
//LODPLUS2 EXEC PGM=AMUUMAIN,  
//PARM='&SSID,AMUEXEMPL,NEW,,MSGLEVEL(1),AMU$OPTS'  
//SSID,UTILID,RSTART,NOTIFY,MSGLEVEL  
//STEPLIB DD DISP=SHR,DSN=product.libraries  
//DD DISP=SHR,DSN=DB2.DSNEXIT  
//DD DISP=SHR,DSN=DB2.DSNLOAD  
//SYSPRINT DD SYSOUT=*  
//UTPRINT DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
```
Example 18 LOAD REPLACE enforcing referential constraints

The following figure shows the SYSPRINT output for example 18:

```
** BM C LOADPLUS FOR DB2 V11R2.00 **
(C) COPYRIGHT 1990 - 2015 BMC SOFTWARE, INC.
```

**SYSPRINT for example 18**

The following figure shows the SYSPRINT output for example 18:
Example 18 LOAD REPLACE enforcing referential constraints

LOADPLUS TECHNOLOGY IS PROTECTED BY U.S. PATENT NUMBERS 7,664,790 AND 7,930,291

LOADPLUS for DB2 Reference Manual
Example 18 LOAD REPLACE enforcing referential constraints
Example 18 LOAD REPLACE enforcing referential constraints

LOADPLUS for DB2 Reference Manual
This example illustrates the use of FASTSWITCH processing during a LOAD REPLACE SRLEVEL REFERENCE load job.

The load is run on a multi-table, segmented table space where each table has one unique clustering index and one non-unique, nonclustering index.
This example uses the FASTSWITCH YES command option to override the FASTSWITCH=NO installation option. LOADPLUS uses the FASTSWITCH process instead of the data set rename process to handle the staging data sets. The SWITCHTIME NOW option tells LOADPLUS to begin the FASTSWITCH process as soon as the load process is complete. In the SYSPRINT, messages BMC50894I and BMC50895I indicate the start and end of the FASTSWITCH process.

The PRELOAD=LOAD default value in the installation options module tells LOADPLUS to use single-phase processing.

Dynamic data set allocation is enabled for most data sets in the installation options module. Additionally, the JCL activates dynamic allocation for sort work data sets by specifying ACTIVE YES for DDTYPE SORTWORK. Specifying ENUMROWS 5000 overrides the AUTOENUMROWS=YES installation option and estimates that LOADPLUS will load 5000 rows.

Because the LOAD command includes the ORDER YES option, LOADPLUS sorts the data rows and clustering index keys.

RULES=BMC is specified in the options module so that the job can use extended operators, such as greater than (>) and less than (<), on the LOAD command.

Message BMC51524I in the SYSPRINT output indicates that conversion errors occurred. Because the EXEC statement designates MSGLEVEL(1), LOADPLUS issues message BMC51501E to identify the erroneous input fields.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

COPY YES INLINE NO REGISTER ALL tells LOADPLUS to create a standard image copy of the loaded table space and register it with DB2.

The following figure shows the JCL for example 19:

```plaintext
//LODPLUS1 EXEC PGM=AMUUMAIN,  // PARM='&SSID,AMUEXA19,NEW,,MSGLEVEL(1),AMU$OPTS'  // * SSID,UTILID,RSTART,NOTIFY,MSGLEVEL  //STEPLIB DD DISP=SHR,DSN=product.libraries  // DD DISP=SHR,DSN=DB2.DSNEXIT  // DD DISP=SHR,DSN=DB2.DSNLOAD  //SYSPRINT DD SYSOUT=*  //SYSOUT DD SYSOUT=*  //UTPRINT DD SYSOUT=*  //SYSUDUMP DD SYSOUT=*  //SYSSREC DD DSN=AMU.QA.RGRTESTS.DTA052,DISP=SHR  //BMCCPY DD DSN=RDAMZL.AMUEXA19.COPYBMC,DISP=(NEW,CATLG,DELETE),  // SPACE=(CYL,(9,9)),UNIT=WORK  //SYSSIN DD *  LOAD DATA REPLACE  SHRLEVEL REFERENCE  FASTSWITCH YES SWITCHTIME NOW
```
SYSPRINT for example 19

The following figure shows the SYSPRINT output for example 19:
Example 19 LOAD REPLACE SHRLEVEL REFERENCE using FASTSWITCH
Example 19 LOAD REPLACE SHRLEVEL REFERENCE using FASTSWITCH

Chapter 5  Examples of LOADPLUS jobs  625
Example 20 LOAD RESUME YES copying multiple partitions to a single copy data set

This example illustrates using a single copy data set to contain a copy of a subset of partitions in a partitioned table space.

In this example, LOADPLUS loads five of the six partitions in the table space.

To enable the subset copy functionality, this example uses a different installation options module to specify COPYSUBSET=YES, and the INTO PART command option specifies a list of contiguous partitions. This job activates dynamic data set allocation for local and remote primary and backup copy data sets (as well as all work files). To enable the single copy for multiple partitions when the copy data sets are dynamically allocated, the installation options module specifies COPYLVL=FULL.

The installation options module includes AUTOENUMROWS=YES, so the command statement does not need to include the ENUMROWS option to enable dynamic allocation. Message BMC51561I shows the number of input records that the utility calculated using automated file size estimation. The LOAD command also includes the DELETEFILES YES SYSDISC YES option to delete the work files, including SYSDISC if it is empty, after the load completes.

In this example, the SYSREC ddname is overridden by using the INDDN command option and specifying the override ddname SYSREC00 in the JCL.

The PRELOAD=LOAD default value in the installation options module tells LOADPLUS to use single-phase processing.

RULES=BMC is specified in the options module so that the job can use extended operators, such as greater than (>) and less than (<), on the LOAD command.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified. LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.
LOADPLUS updates statistics in the DB2 catalog. The JCL for this example includes
the DD statement //ASUSRPRRT DD DUMMY. This DD statement tells LOADPLUS
not to produce a report of the statistics.

The following figure shows the JCL for example 20:

**JCL for example 20**

```plaintext
//BMCLDRA1 EXEC PGM=AMUUMAIN,
  PARM='&SSID,AMUEXP20,NEW,,MSGLEVEL(1),AMU$OPTP'
//**
//STEPLIB DD DISP=SHR,DSN=product.libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD SYSOUT=* 
//SYSPUT DD SYSOUT=* 
//SYSDUMPP DD SYSOUT=* 
//ASUSRPRRT DD DUMMY
//SYSREC00 DD DSN=AMU.RDAWQP.UNLOAD1.AMULIDA,DISP=SHR
//SYSIN DD *
LOAD DATA RESUME YES UPDATEDB2STATS YES
ORDER YES
UNIQUECHECK NO
COPY YES INLINE NO REGISTER ALL
INDDN SYSREC00
DELETEFILES YES SYSDISC YES
INTO TABLE AMU910.AMUSSIAT PART 5,3,1,2,4 REPLACE
(EBDW_IP_ADDR_NM POSITION(1:15) CHAR (15)
,EBDW_SESS_ID POSITION(16:30) CHAR (15)
,EBDW_END_TS POSITION(31:56) TIMESTAMP EXTERNAL(26)
,EBDW_END_DT POSITION(57:66) DATE EXTERNAL(10)
,EBDW_END_TIME POSITION(67:74) CHAR (8)
,EBDW_END_HOUR POSITION(75:76) SMALLINT
,EBDW_START_TS POSITION(77:102) TIMESTAMP EXTERNAL(26)
,EBDW_START_DT POSITION(103:112) DATE EXTERNAL(10)
,EBDW_START_TIME POSITION(113:120) CHAR (8)
,EBDW_LOG_TYP_CD POSITION(121:122) SMALLINT
,EBDW_LOG_TYP_TXT POSITION(123:148) CHAR (25)
,EBDW_EVNT_CD POSITION(149:163) CHAR (15)
,EBDW_SBU_NM POSITION(164:168) CHAR (5)
,EBDW_APP_NM POSITION(169:178) CHAR (10)
,EBDW_USR_ID POSITION(179:188) CHAR (10)
,EBDW_SURR_ID POSITION(189:198) CHAR (10)
,EBDW_SS_NMBR POSITION(199:207) CHAR (9)
,EBDW_PAGE_NM POSITION(208:257) CHAR (50)
,EBDW_BROWER_NM POSITION(258:332) CHAR (75)
,EBDW_TRNS_NM POSITION(333:377) CHAR (50)
,EBDW_DRTN_SCND POSITION(378:381) INTEGER
,EBDW_SYS_NMBR POSITION(382:386) SMALLINT
NULLIF BMC_NULL1=X'6F'
,EBDW_SYS_ADMIN_ID POSITION(387:407) CHAR (30)
NULLIF BMC_NULL2=X'6F'
,EBDW_MEMBER POSITION(387:388) INTEGER
NULLIF BMC_NULL3=X'6F'
,EBDW_SYS_MEMBER POSITION(387:388) INTEGER
NULLIF BMC_NULL4=X'6F'
,EBDW_TRK_ID POSITION(396:427) CHAR (32)
,EBDW_TRNS_NM POSITION(428:477) CHAR (50)
,EBDW_DRTN_SCND POSITION(478:481) INTEGER
```

Chapter 5 Examples of LOADPLUS jobs  627
Example 20 LOAD RESUME YES copying multiple partitions to a single copy data set

The following figure shows the SYSPRINT output for example 20:
Example 20 LOAD RESUME YES copying multiple partitions to a single copy data set
Example 20 LOAD RESUME YES copying multiple partitions to a single copy data set
Example 20 LOAD RESUME YES copying multiple partitions to a single copy data set
Example 21 Loading LOB and XML data

This example is a LOAD REPLACE of a partitioned table space that contains LOB and XML columns.

The table space has the following indexes:

- A non-unique clustering index
- A unique index on the row ID column
- A document ID index (because the table contains an XML column)

Also participating in this job are the following table spaces:

- Four LOB table spaces and their auxiliary indexes
- An implicitly created XML table space and its node ID index

This job loads LOB and XML data from referenced files. The SYSREC in this example shows the referenced file names, and the field specification shows the use of BLOBF and CLOBF with the CHAR data type for the LOB and XML data.

Dynamic allocation is active for primary copies of the base table space, the LOB table spaces, and the XML table space. LOADPLUS produces a copy for each table space, and COPYLVL FULL tells LOADPLUS to produce a single copy for all partitions.

Message 50041I indicates the status of zIIP processing. For this example, the ZIIP installation option is ENABLED, but an XBM subsystem was not specified.
LOADPLUS automatically located an available XBM subsystem to provide zIIP processing.

The following figure shows the DDL for example 21:

```
CREATE TABLESPACE S2UDSSL4 IN D2UDSSL4
  USING STOGROUP SGLLOBV9
  COMPRESS NO
  BUFFERPOOL BP32K
  NUMPARTS 2;
  COMMIT;

CREATE TABLE AMU.TABLE_AMU_MIXED_NO_TABLE_PARTITION
  (COL_INTEGER     INTEGER,
   COL2_CLOB       CLOB(10M) FOR SBCS DATA ,
   COL3_BLOB       BLOB(10M),
   COL4_XML        XML,
   COL5_ROWID      ROWID NOT NULL GENERATED ALWAYS
  )
  IN D2UDSSL4.S2UDSSL4
  PARTITION BY
  ( COL_INTEGER )
  (PARTITION 1 ENDING AT (3500),
   PARTITION 2 ENDING AT (2147483646)
  );

CREATE INDEX AMU.IXLOBMN4
  ON AMU.TABLE_AMU_MIXED_NO_TABLE_PARTITION
  ( COL_INTEGER )
  USING STOGROUP SYSDEFLT
  PRIQTY 1200
  SECQTY 1200
  ERASE NO
  FREEPAGE 0
  PCTFREE 5
  GBPCACHE CHANGED
  BUFFERPOOL BP0
  CLOSE YES
  CLUSTER;
  COMMIT;

CREATE UNIQUE INDEX AMU.IXLOBRI4
  ON AMU.TABLE_AMU_MIXED_NO_TABLE_PARTITION
  ( COL5_ROWID ) ;
  COMMIT;

CREATE LOB TABLESPACE SLLBS401 IN D2UDSSL4
  USING STOGROUP SGLLOBV9
  BUFFERPOOL BP0
  DSSIZE 1G;
  COMMIT;

CREATE AUX TABLE AMU.AUXTABLE_FOR_CLOB_AMU_MIXED_NO_TABLE_PARTITION_1
  IN D2UDSSL4.SLLBS401
  STORES AMU.TABLE_AMU_MIXED_NO_TABLE_PARTITION
  COLUMN COL2_CLOB PART 1;
  COMMIT;

CREATE UNIQUE INDEX AMU.IXD2L401
  ON AMU.AUXTABLE_FOR_CLOB_AMU_MIXED_NO_TABLE_PARTITION_1
  USING STOGROUP SGLLOBV9;
  COMMIT;

CREATE LOB TABLESPACE SLLBS402 IN D2UDSSL4
  USING STOGROUP SGLLOBV9
```
The following figure shows the JCL for example 21:

```sql
BUFFERPOOL BP16K0
DSSIZE 2G;
COMMIT;

CREATE AUX TABLE AMU.AUXTABLE_FOR_CLOB_AMU_MIXED_NO_TABLE_PARTITION_2
  IN D2UDDL4.SLLBS402
  STORES AMU.TABLE_AMU_MIXED_NO_TABLE_PARTITION
  COLUMN COL2_CLOB PART 2;
COMMIT;

CREATE UNIQUE INDEX AMU.IXD2L402
  ON AMU.AUXTABLE_FOR_CLOB_AMU_MIXED_NO_TABLE_PARTITION_2
  USING STOGROUP SGLLOBV9;
COMMIT;

CREATE LOB TABLESPACE SLLBS411 IN D2UDDL4
  USING STOGROUP SGLLOBV9
  BUFFERPOOL BP16K0
  DSSIZE 8G;
COMMIT;

CREATE AUX TABLE AMU.AUXTABLE_FOR_BLOB_AMU_MIXED_NO_TABLE_PARTITION_1
  IN D2UDDL4.SLLBS411
  STORES AMU.TABLE_AMU_MIXED_NO_TABLE_PARTITION
  COLUMN COL3_BLOB PART 1;
COMMIT;

CREATE UNIQUE INDEX AMU.IXD2L411
  ON AMU.AUXTABLE_FOR_BLOB_AMU_MIXED_NO_TABLE_PARTITION_1
  USING STOGROUP SGLLOBV9;
COMMIT;

CREATE LOB TABLESPACE SLLBS412 IN D2UDDL4
  USING STOGROUP SGLLOBV9
  BUFFERPOOL BP16K0
  DSSIZE 16G;
COMMIT;

CREATE AUX TABLE AMU.AUXTABLE_FOR_BLOB_AMU_MIXED_NO_TABLE_PARTITION_2
  IN D2UDDL4.SLLBS412
  STORES AMU.TABLE_AMU_MIXED_NO_TABLE_PARTITION
  COLUMN COL3_BLOB PART 2;
COMMIT;

CREATE UNIQUE INDEX AMU.IXD2L412
  ON AMU.AUXTABLE_FOR_BLOB_AMU_MIXED_NO_TABLE_PARTITION_2
  USING STOGROUP SGLLOBV9;
COMMIT;
```

The following figure shows the JCL for example 21:

```sql
//BMCLOAD1 EXEC PGM=AMUUMAIN,
  //           PARM='&SSID,AMUEX21,NEW,,MSGLEVEL(1),AMU$OPTS'
//STEPLIB  DD DISP=SHR,DSN=product.libraries
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//         DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD SYSOUT=*;
//UTPRINT  DD SYSOUT=*;
//SYSUDUMP DD UNIT=WORK,SPACE=(CYL,(20,10)),DISP=(NEW,DELETE)
//SORTOUT DD UNIT=WORK,SPACE=(CYL,(20,10)),DISP=(NEW,PASS)
//SORTWK01 DD UNIT=WORK,SPACE=(CYL,(20,10)),DISP=(NEW,PASS)
//SORTWK02 DD UNIT=WORK,SPACE=(CYL,(20,10)),DISP=(NEW,PASS)
//SORTWK03 DD UNIT=WORK,SPACE=(CYL,(20,10)),DISP=(NEW,PASS)
//SORTWK04 DD UNIT=WORK,SPACE=(CYL,(20,10)),DISP=(NEW,PASS)
//SYSERR DD UNIT=WORK,SPACE=(CYL,(1,1)),DISP=(NEW,DELETE)
```
 SYSREC for example 21

The following figure shows a portion of the SYSREC for example 21:

SYSPRINT for example 21

The following figure shows the SYSPRINT output for example 21:
Example 21 Loading LOB and XML data

LOADPLUS for DB2 Reference Manual
Example 21 Loading LOB and XML data

Chapter 5  Examples of LOADPLUS jobs

Example 21 Loading LOB and XML data
LOAD PLUS for DB2 Reference Manual
Chapter 5  Examples of LOADPLUS jobs  639

Example 21 Loading LOB and XML data
Example 22 Running a DSNUTILB load

This example illustrates requirements and output when you use a feature that requires LOADPLUS to invoke DSNUTILB.

As shown in this example, LOADPLUS displays the DSN messages, including the TEMPLATE statements, in the job SYSPRINT.

In this case, LOADPLUS is loading a partitioned table space that has a unique index that contains keys with random ordering. The options module specifies YES for the DSNUTILB installation option.

The LOAD command includes DDTYPE statements with ACTIVE YES for all work files and for the copy data sets needed for this job. ACTIVE YES enables dynamic allocation for these data sets as required for a DSNUTILB load.

The return code 4 from DSNUTILB is passed on to LOADPLUS as the completion code for the load job.

The following figure shows the JCL for example 22:

```
//RESUME EXEC PGM=AMUUMAIN,
  // PARM='&SSID,AMUEX22,NEW,,MSGLEVEL(1),AMU$OPTI'
//STEPLIB DD DISP=SHR,DSN=product.Library
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//         DD DISP=SHR,DSN=DB2.DSNLOAD
//SYSPRINT DD SYSOUT=*  // Load the user exit library.
//SYSOUT DD SYSOUT=*  // Load the user exit library.
//UTPRINT DD SYSOUT=*  // Load the user exit library.
//SYSUDUMP DD SYSOUT=*  // Load the user exit library.
//SYSREC DD DISP=SHR,DSN=AMU.VQA920.FEATURES.V552292.CLOB1
//SYRIN DD *  // Load the user exit library.
LOAD DATA INDDN SYSREC RESUME YES
COPY NO COPYPEND NO
INTO TABLE AMU.TBIXE102
  ( DSNULL_IND_00001 POSITION( 00001) CHAR(1)
    , COL_SMALL
      POSITION( 00002:00003) SMALLINT
        NULLIF DSNULL_IND_00001 =X'FF'
    , DSNULL_IND_00002 POSITION( 00004) CHAR(1)
    , COL_INTEGER
      POSITION( 00005:00008) INTEGER
        NULLIF DSNULL_IND_00002 =X'FF'
    , DSNULL_IND_00003 POSITION( 00009) CHAR(1)
    , COL_FLOAT
      POSITION( 00010:00017) FLOAT(53)
        NULLIF DSNULL_IND_00003 =X'FF'
    , DSNULL_IND_00004 POSITION( 00018) CHAR(1)
    , COL_CHAR
      POSITION( 00019:00028) CHAR(00010)
        NULLIF DSNULL_IND_00004 =X'FF'
    , DSNULL_IND_00005 POSITION( 00029) CHAR(1)
    , COL_VCHAR
      POSITION( 00030) VARCHAR
        NULLIF DSNULL_IND_00005 =X'FF'
    , DSNULL_IND_00006 POSITION( *) CHAR(1)
    , COL_DATE
      POSITION( *) DATE EXTERNAL
```
Example 22 Running a DSNUTILB load

SYSPRINT for example 22

The following figure shows the SYSPRINT output for example 22:
Chapter 5  Examples of LOADPLUS jobs

Example 22 Running a DSNUTILB load
Tuning LOADPLUS jobs

This chapter provides information that enables you to improve the performance or memory use of your jobs.

Tuning for performance in LOADPLUS

The LOADPLUS for DB2 product provides several options that you can use to influence its performance.

Some options affect the performance of the entire product; others influence processing only for specific functions of LOADPLUS. This topic in this section explains the effect that these options have on the performance of LOADPLUS.

Setting installation options for optimal performance in LOADPLUS

In general, the values that were shipped with LOADPLUS for the installation options provide you with optimal performance. However, for certain environments, BMC recommends that you make adjustments to these values.

The following table describes these recommendations.

---

**Note**

If you modify these options after installation, you must rerun the installation job for these modifications to take effect.
### Table 111: Installation option changes for performance

<table>
<thead>
<tr>
<th>Installation option</th>
<th>Recommended value</th>
<th>Comments</th>
<th>Reference to additional performance information</th>
</tr>
</thead>
<tbody>
<tr>
<td>COPYLVL</td>
<td>PART</td>
<td>Consider this value if you are loading partitioned table spaces and you want LOADPLUS to make copies. Specifying COPYLVL PART tells LOADPLUS to make copies by partition. <strong>Note:</strong> You might need to specify FULL when you have to use a tape device for copies. Additionally, FULL is required when LOADPLUS invokes DSNUTILB.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>DDTYPE</td>
<td>ACTIVE YES for the data sets to dynamically allocate</td>
<td>BMC recommends that you dynamically allocate the copy data sets and SYSUT1 and SORTOUT files for LOADPLUS. <strong>Note:</strong> BMC recommends that you do not dynamically allocate sort work files through LOADPLUS, but allow BMCSORT to allocate them by specifying a value greater than 0 for the SORTNUM installation or command option.</td>
<td>“Performance considerations for dynamic allocation options” on page 659</td>
</tr>
<tr>
<td>Installation option</td>
<td>Recommended value</td>
<td>Comments</td>
<td>Reference to additional performance information</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------</td>
<td>----------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>DELFILES</td>
<td>(YES,YES), if possible</td>
<td>LOADPLUS deletes the work files only after the load is complete and, for the second value, if no discards were written to the discard file. Many jobs include a cleanup step that follows a LOADPLUS step. Specifying (YES,YES) or (YES,NO) eliminates the need to change this cleanup JCL.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>DRNDELAY, DRNRETRY, and FORCE</td>
<td></td>
<td>Adjust these options based on how long your application can wait for a successful drain.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>KEEPDICTIONARY</td>
<td>YES</td>
<td>Consider this value to save building a new dictionary if one already exists.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>REDEFINE</td>
<td>NO</td>
<td>This value creates additional data sets only when needed and cleans up unused data sets. Specify this value if the objects do not require a change in their size. This option can improve performance and is particularly useful when you are loading table spaces with many tables and indexes (for example in an ERP environment) or when you are loading many partitions. If you are running a load job that uses staging data sets, you can improve the performance of the job by specifying REDEFINE NO and preallocating the staging data sets.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Installation option</td>
<td>Recommended value</td>
<td>Comments</td>
<td>Reference to additional performance information</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------</td>
<td>----------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>SMCORE</td>
<td>(0K,0K)</td>
<td>This value is the default value. BMC recommends that you do not change this value except for the situation described in the referenced section or under instructions from BMC Customer Support.</td>
<td>“Performance considerations for the sort processing options” on page 665</td>
</tr>
<tr>
<td>STOPRETRY</td>
<td>450</td>
<td>If you plan to load objects with more than 2000 partitions, specify a minimum value of 450.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>UXSTATE</td>
<td>PROB</td>
<td>If you can ensure that all exits that LOADPLUS calls can run in problem states, specify this value to obtain significant savings in CPU overhead.</td>
<td>“Performance considerations for the UXSTATE installation option” on page 668</td>
</tr>
<tr>
<td>WBUFFS</td>
<td>(20,20)</td>
<td>If you are running with REGION=0M, or you have 64 MB of virtual storage available above the line, specify a value of 20 for the second parameter.</td>
<td>“Performance considerations for buffer options” on page 656</td>
</tr>
</tbody>
</table>

**Specifying command options for optimal performance in LOADPLUS**

The tables in this topic describe command option recommendations to obtain optimal performance when running LOADPLUS.

*Note*

Some of these recommendations apply only to specific environments or conditions. Review the information in these tables to determine whether the recommendations meet your needs.

The following table lists LOAD command options that correspond to the installation options listed in Table 111 on page 646. BMC recommends that you specify the LOAD command options that are listed in this table if they are not already specified in the installation options. See Table 111 on page 646 for comments about these command options.
### Table 112: Command option values for performance that correspond to installation options

<table>
<thead>
<tr>
<th>Command option</th>
<th>Corresponding installation option</th>
</tr>
</thead>
<tbody>
<tr>
<td>COPYLVL PART</td>
<td>COPYLVL=PART</td>
</tr>
<tr>
<td>DDTYPE ACTIVE YES for copy, SYSUT1, and SORTOUT data sets</td>
<td>DDTYPE ACTIVE YES for copy, SYSUT1, and SORTOUT data sets</td>
</tr>
<tr>
<td>DELETEFILES YES SYSDISC YES</td>
<td>DELFILES=(YES,YES)</td>
</tr>
<tr>
<td>KEEPDICTIONARY YES</td>
<td>KEEPDICTIONARY=YES</td>
</tr>
<tr>
<td>REDENDFNE NO</td>
<td>REDENDFNE=NO</td>
</tr>
</tbody>
</table>

The following table describes additional recommended command options:

### Table 113: Additional command option values for performance

<table>
<thead>
<tr>
<th>Command option</th>
<th>Conditions and comments</th>
<th>Reference to additional performance information</th>
</tr>
</thead>
<tbody>
<tr>
<td>APCOMMIT and APMAXAGENTS</td>
<td>If you are running an SQLAPPLY load, low cardinality keys exist, and there is high CPU usage, the values for APCOMMIT and APMAXAGENTS can impact performance. Adjust one parameter at a time to determine the best combination of values. <strong>Note:</strong> APMAXAGENTS affects only partitioned or multi-table table spaces. LOADPLUS uses one agent per partition or table. For partition-by-growth table spaces, LOADPLUS uses only one agent per load regardless of the value that you specify for APMAXAGENTS.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>CCSID</td>
<td>Do not specify this option unless your job requires it. CCSID translation can degrade the performance of the load job.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Command option</td>
<td>Conditions and comments</td>
<td>Reference to additional performance information</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ENUMROWS</td>
<td>BMC recommends that you enable automated file size estimation by either using AUTOENUMROWS=YES (the default) in your installation options or by specifying ENUMROWS AUTO on the LOAD command. Except for a DSNUTILB load, dynamic allocation requires either automated file size estimation or an appropriate numeric value for the ENUMROWS command option.</td>
<td>“Performance considerations for dynamic allocation options” on page 659</td>
</tr>
<tr>
<td>FORMAT BMCUNLOAD</td>
<td>Specify this option if you are performing data migration between DB2 objects with identical or almost-identical definitions and both LOADPLUS and UNLOAD PLUS are installed. Use this option in conjunction with the FORMAT BMCLOAD option in UNLOAD PLUS.</td>
<td>“Loading data from UNLOAD PLUS (FORMAT BMCUNLOAD option)” on page 675</td>
</tr>
<tr>
<td>INDEX BUILD</td>
<td>Specify INDEX BUILD when possible. This option is usually faster than INDEX UPDATE and results in a fully reorganized index. <strong>Note:</strong> INDEX BUILD requires that there be enough memory and DASD to build the index. If you must specify INDEX UPDATE, try to ensure that the index is very well organized and has plenty of free space. BMC recommends that you specify INDEX UPDATE only when running a LOAD RESUME YES with a small number of updates.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>INTO and the PART option</td>
<td>Specify the INTO statement with the PART option using the guidelines described in the referenced topic.</td>
<td>“INTO and PART” on page 660</td>
</tr>
</tbody>
</table>
ORDER PRESORTED
If your data is already sorted and you are using a single input file, specify ORDER PRESORTED.
“ORDER PRESORTED” on page 662

ORDER YES
If you are loading a partitioned table space (except partition-by-growth), consider specifying ORDER YES.
“ORDER YES” on page 661

UNIQUEINTO YES
Specify this option if you have multiple INTO specifications.
“UNIQUEINTO YES” on page 660

### Additional performance information for LOADPLUS installation and command options

The following topics provide performance information, in addition to the information in the previous tables, for some of the installation and command options.

This information can help you make decisions about these options that can affect the performance of your load jobs. Use the following table to find information about a specific option:

<table>
<thead>
<tr>
<th>Option</th>
<th>Controls</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANALYZE</td>
<td>Provides information to help you determine data set sizes and the optimal number of tasks</td>
<td>“Performance considerations for ANALYZE command options” on page 653</td>
</tr>
<tr>
<td>APMULTIROW</td>
<td>Controls the number of rows inserted during a single insert action when running an SQLAPPLY load</td>
<td>“Performance considerations for APMULTIROW command and installation options” on page 655</td>
</tr>
<tr>
<td>AUTOENUMROWS</td>
<td>Automates file size estimation for dynamic allocation and optimal sort processing</td>
<td>“Performance considerations for dynamic allocation options” on page 659</td>
</tr>
<tr>
<td>CBUFFS</td>
<td>Controls buffers for writing copy data sets</td>
<td>“Performance considerations for buffer options” on page 656</td>
</tr>
<tr>
<td>DDTYPE</td>
<td>Controls dynamic allocation</td>
<td>“Performance considerations for dynamic allocation options” on page 659</td>
</tr>
<tr>
<td>Option</td>
<td>Controls</td>
<td>Reference</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DISCARDS IGNORE</td>
<td>Ignores discarded data</td>
<td>“Performance considerations for the DISCARDS IGNORE command option” on page 658</td>
</tr>
<tr>
<td>ENUMROWS</td>
<td>Provides information about input file size for dynamic allocation</td>
<td>“Performance considerations for dynamic allocation options” on page 659</td>
</tr>
<tr>
<td>IBUFFS</td>
<td>Controls buffers for reading SYSREC data sets</td>
<td>“Performance considerations for buffer options” on page 656</td>
</tr>
<tr>
<td>INTO</td>
<td>Controls which input records are loaded</td>
<td>“Performance considerations for the INTO and UNIQUEINTO command options” on page 660</td>
</tr>
<tr>
<td>LBUFFS</td>
<td>Controls buffers for reading and writing SORTOUT data sets</td>
<td>“Performance considerations for buffer options” on page 656</td>
</tr>
<tr>
<td>MAXP</td>
<td>Controls concurrent partition access</td>
<td>“Performance considerations for the MAXP installation option” on page 661</td>
</tr>
<tr>
<td>MAXSORTS</td>
<td>Controls the number of certain types of tasks</td>
<td>“Performance considerations for the sort processing options” on page 665</td>
</tr>
<tr>
<td>ORDER</td>
<td>Provides information about whether the data is sorted</td>
<td>“Performance considerations for the ORDER command option” on page 661</td>
</tr>
<tr>
<td>PRELOAD ANALYZE</td>
<td>Provides information to help you optimize the number of tasks</td>
<td>“Performance considerations for the PRELOAD ANALYZE command option” on page 662</td>
</tr>
<tr>
<td>PRELOAD LOAD</td>
<td>Performs single-phase load</td>
<td>“Performance considerations for the PRELOAD LOAD installation and command options” on page 663</td>
</tr>
<tr>
<td>REPLACE</td>
<td>Replaces existing data in the table space</td>
<td>“Performance considerations for the RESUME and REPLACE command options” on page 664</td>
</tr>
<tr>
<td>RESUME</td>
<td>Adds data into table spaces that already contain data</td>
<td>“Performance considerations for the RESUME and REPLACE command options” on page 664</td>
</tr>
<tr>
<td>RULES</td>
<td>Controls data comparison processing</td>
<td>“Performance considerations for the RULES installation option” on page 665</td>
</tr>
<tr>
<td>SMAX</td>
<td>Controls the number of certain types of tasks</td>
<td>“Performance considerations for the sort processing options” on page 665</td>
</tr>
</tbody>
</table>
Performance considerations for ANALYZE command options

The ANALYZE options limit abends that are caused by inadequate size allocations for data sets because they provide input to dynamic data set allocation processing or provide information that allows you to allocate space more accurately.

LOADPLUS uses the results of the ANALYZE phase to verify data set sizes and optimize tasks.

ANALYZE

If you do not specify ANALYZE or if you specify ANALYZE without SAMPLE or SCAN, LOADPLUS determines whether to sample or scan for cardinality. LOADPLUS always obtains average row length information from the DB2 catalog.

ANALYZE SAMPLE and SCAN

If you specify ANALYZE SAMPLE, LOADPLUS samples the best index on each participating table to estimate the cardinality. If you specify ANALYZE SCAN, LOADPLUS scans the best index on each participating table to find the exact cardinality. For ANALYZE SAMPLE or SCAN, LOADPLUS always obtains the average row length information from the DB2 catalog.
The following table shows how the SCAN and SAMPLE options affect the elapsed time of the load and the accuracy of the data set sizings. Determine which option to select based on the needs of your organization.

Table 115: Time vs. accuracy for the ANALYZE options

<table>
<thead>
<tr>
<th>Option</th>
<th>Time required</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCAN</td>
<td>More</td>
<td>More</td>
</tr>
<tr>
<td>SAMPLE</td>
<td>Less</td>
<td>Less</td>
</tr>
</tbody>
</table>

**ANALYZE PAUSE and ONLY**

The ANALYZE PAUSE and ANALYZE ONLY options provide estimates of the space that LOADPLUS needs for the following data sets:

- Load (SORTOUT)
- Work (SYSUT1)
- Sort work (SORTWK)
- Discard (SYSDISC),
- Error (SYSERR)
- Image copy (BMCCPY, BMCCPZ, BMCRCY, and BMCRCZ)

*Note*

ANALYZE does not provide estimates for LOB or XML copy data sets. These data sets must be dynamically allocated.

LOADPLUS writes these statistics to SYSPRINT as shown in the following messages. For more information about using this report for allocating your data sets, see “ANALYZE option for estimating data set allocation in LOADPLUS” on page 465.

LOADPLUS issues a separate BMC51533I message for each data set and provides the following information:

- Data set name
- Number of kilobytes
- Primary and secondary 3380 cylinder quantities
- Primary and secondary 3390 cylinder quantities
- Index name, where applicable
Multiple BMC51533I messages provide the estimated values. Because some types of load jobs allow you to specify single or multiple SORTOUT and SYSUT1 data sets, LOADPLUS provides values for both.

**Additional ANALYZE-related messages**

The following messages (BMC50485I, BMC51535I, and BMC51536I) provide estimates that you can use to gauge the elapsed time of the load:

**BMC50485I**

```
ESTIMATED CARDINALITY OF [PART | TABLE] [partitionNumber | tableName] = n
```

This message provides either an estimate or the exact cardinality of each partition or table that is involved in the load. LOADPLUS issues this message for either an ANALYZE SAMPLE or an ANALYZE SCAN. You can use these values to gauge the elapsed time of the load.

**BMC51535I**

```
ESTIMATED CARDINALITY OF SPACE = n
ESTIMATED AVG ROW LENGTH = n
```

This message provides an estimate of the cardinality and of the average row length of the table space. LOADPLUS issues this message when performing an ANALYZE SAMPLE. You can compare these values with estimates that other messages provide and with previous executions of the utility to validate the accuracy of these values and to gauge the elapsed time of the load.

**BMC51536I**

```
CARDINALITY OF SPACE = n
ESTIMATED AVG ROW LENGTH = n
```

This message provides the exact cardinality and an estimate of the average row length of the table space. LOADPLUS issues this message when it performs an ANALYZE SCAN. You can compare these values with estimates that other messages provide and with previous executions of the utility to validate the accuracy of these values and to gauge the elapsed time of the load.

**Performance considerations for APMULTIROW command and installation options**

The APMULTIROW option controls the number of rows that are inserted in a single insert action when running an SQLAPPLY load.

In most cases, BMC recommends a value of 100 (the value that is shipped with LOADPLUS).

The performance benefits of multiple-row insert actions depend on conditions in your environment. BMC recommends testing various values for this option in a test environment before changing the value in a production environment.
Performance considerations for buffer options

You can use installation options to control the amount of buffer space that the data sets use during LOADPLUS processing.

For each option, LOADPLUS multiplies the number that you specify by a particular value (explained in the following sections) to determine the amount of virtual storage needed for the buffer pool. LOADPLUS then allocates as many buffers as will fit into that space.

LOADPLUS allocates all VSAM buffers and sequential I/O buffers above the 16-megabyte line.

SYSREC data set

The SYSREC data set contains the input data that you are loading. LOADPLUS reads this data set at least once during the PRELOAD phase or COMBINED phase to build DB2 data rows for loading. LOADPLUS also reads the data set during discard processing if any records are discarded.

LOADPLUS uses the IBUFFS value to determine the amount of buffer pool storage to allocate for reading each active SYSREC data set. The multiplier that LOADPLUS applies to the IBUFFS option value varies as follows:

- **For SYSREC data sets with a block size greater than 32 KB**, LOADPLUS uses a multiplier of 256 KB. In this case, if the IBUFFS option has a value greater than 6, LOADPLUS ignores that setting and uses 6.
- **For all other SYSREC data sets**, LOADPLUS uses a multiplier of 32 KB. In this case, BMC recommends a value of 20 for the IBUFFS option, which is the value that is shipped with LOADPLUS.

SORTOUT data sets

For a two-phase load job, the SORTOUT data set contains the DB2 row images for loading in the LOAD phase. LOADPLUS writes this data set in the PRELOAD phase and reads it in the LOAD phase.

For most single-phase load jobs, LOADPLUS does not use the SORTOUT data set. However, for those single-phase jobs that do use it, the SORTOUT data set contains the data-sorting index keys and the DB2 row images. LOADPLUS writes this data set in the COMBINED phase and reads it only if you restart your job. For more information, see “SORTOUT data sets in LOADPLUS” on page 443.

If one or more of the tables that you are loading uses an EDITPROC routine for data compression, the data is written in its compressed format to the SORTOUT data set.
Using compressed data improves performance considerably because LOADPLUS requires fewer I/O operations to process the data.

LOADPLUS uses the LBUFFS installation option to determine the amount of buffer pool storage that is allocated for reading and writing each active SORTOUT data set. The multiplier that LOADPLUS applies to the LBUFFS option value is 32 KB. For the LBUFFS option, BMC recommends a value of 20 (the value that is shipped with LOADPLUS). LOADPLUS determines the optimal block size of the SORTOUT data set, depending on the device type used.

**SYSUT1 data sets**

The SYSUT1 data set contains the information that LOADPLUS needs to build all indexes (when you specify ORDER NO) or only the non-data-sorting indexes (when you specify ORDER YES). For a two-phase load job, LOADPLUS writes to this data set during the PRELOAD phase and reads it during the LOAD phase. For most single-phase load jobs, LOADPLUS does not use the SYSUT1 data set. However, for those single-phase jobs that do use the SYSUT1 data set, LOADPLUS writes this data set in the COMBINED phase and reads it only if you restart your job. For more information, see “SYSUT1 data sets in LOADPLUS” on page 459.

When more than one non-data-sorting index is participating in the load, you can improve I/O performance by using multiple SYSUT1 data sets. Using multiple data sets reduces the number of data blocks that LOADPLUS writes and allows I/O operations to overlap.

When you specify a single SYSUT1 data set for all indexes, the data set requires a record length that is long enough to hold information for the longest key. If shorter keys exist, LOADPLUS pads them so that they are as long as the longest key. With multiple SYSUT1 data sets, LOADPLUS writes information for each index to its own SYSUT1 data set, and does not pad the keys.

LOADPLUS uses the WBUFFS installation option to determine the amount of buffer pool storage that is allocated for reading and writing the SYSUT1 data set. The multiplier that LOADPLUS applies to the WBUFFS option value is 32 KB.

The WBUFFS option has two parameter values. LOADPLUS uses the first number if you specify only one work data set. LOADPLUS uses the second number for each data set if you specify multiple work data sets. BMC recommends the following values for WBUFFS:

- If you run with REGION=0M or you have 64 MB of virtual storage available above the line, specify 20 for a single SYSUT1 data set and 20 for multiple SYSUT1 data sets.

- Otherwise, specify 20 for a single SYSUT1 data set and 10 for multiple SYSUT1 data sets, which are the values that are shipped with LOADPLUS.
LOADPLUS determines the optimal block size of the SYSUT1 data set, depending on the device type containing the data set.

**Note**
For an SQLAPPLY load or a single-phase LOAD REPLACE job, LOADPLUS does not require a SYSUT1 data set and does not use it if you specify it.

### Copy data sets

LOADPLUS writes the copy data sets during the LOAD or COMBINED phase. These data sets contain identical copies of the loaded table space or partitions of the table space after it is loaded. (See “Copy data sets in LOADPLUS” on page 436 for detailed information about the use of the copy data sets in LOADPLUS.)

LOADPLUS uses the CBUFFS installation option to determine the amount of buffer pool storage that is allocated for writing the copy data sets. The multiplier that LOADPLUS applies to the CBUFFS option value is 32 KB. For the CBUFFS option, BMC recommends a value of 30 (which is the value that is shipped with LOADPLUS). If multiple partitions for a copy of the table space are being copied to separate data sets, LOADPLUS uses only one group of buffers.

LOADPLUS determines optimal block size of copy data sets based on the device type that contains the data set. Copy data sets can be on different device types. However, if additional copy data sets for the same object (for example, remote copy data sets) are on different device types, the block size for all copy data sets for that object is the block size that LOADPLUS determined was optimal for the local primary copy.

### DB2 data sets

The DB2 data sets contain the loaded data and associated indexes in a format that is usable by DB2. LOADPLUS reads those data sets that contain the indexes during the PRELOAD or COMBINED phase if you specify the RESUME YES (with or without PART REPLACE) option on the LOAD command.

LOADPLUS assigns buffers for these data sets based on available virtual storage and the number of partitions that LOADPLUS is processing concurrently. Providing as much virtual storage as possible allows LOADPLUS to process multiple partitions and indexes concurrently. For more information, see “Providing maximum virtual storage” on page 673.

### Performance considerations for the DISCARDS IGNORE command option

The DISCARDS IGNORE command option allows you to specify the discard types to ignore.
Except for duplicate key discards, LOADPLUS does not write ignored discards to the SYSERR file, does not report them in the SYSPRINT error summary report, and does not discard them to the discard (SYSDISC) data set. However, if you specify MSGLEVEL(1), LOADPLUS still produces other diagnostic messages associated with discards for debugging purposes.

Using the DISCARDS IGNORE option to ignore WHEN and PART discards can improve performance significantly when you anticipate a high volume of discards.

**Performance considerations for dynamic allocation options**

Although dynamically allocating each data set does require some processing time, enabling dynamic data set allocation can improve performance in many cases.

If you are loading very small quantities of data, the extra processing time might be a nontrivial percentage of the elapsed time. However, if you are loading large quantities of data, especially in a partitioned data set, using dynamic data set allocation might improve performance because it always uses the optimal number of data sets, enabling LOADPLUS to use DASD more efficiently.

In general, the best files to dynamically allocate with the DDTYPE options are SORTOUT and SYSUT1. For details, see “Dynamic data set allocation in LOADPLUS” on page 78, the individual data types in “LOADPLUS DD statements” on page 432, and “Dynamic allocation options” on page 393.

**Note**

Ensure that the DSNPAT pattern for each data set type generates unique data set names. BMC recommends that you use the database and table space names in the DSNPAT pattern to prevent duplicate data set names. If you are loading partitioned table spaces, include the partition number in the pattern name for copy data sets. Using date and time in work data sets is also useful to avoid duplicate data set names.

You can improve performance further with dynamic allocation by enabling automated file size estimation (AUTOENUMROWS=YES or ENUMROWS AUTO), or by providing a numeric value for the ENUMROWS command option that is as accurate as possible. If you specify a numeric value, it does not have to be exact. BMC recommends that you overestimate rather than underestimate. Overestimating results in LOADPLUS allocating files that are larger than needed, but underestimating can cause LOADPLUS to terminate.

If you specify a numeric value for ENUMROWS, the following information provides guidelines for the two most common scenarios:

- If the amount of data can vary significantly, specify a value that is approximately 90% of the largest number of rows that you will load at one time.
If you specify LOAD RESUME YES, use the form ENUMROWS(,new). Again, if the amount of data that you are adding varies, use a number that is approximately 90% of the largest amount that you will add at one time.

Performance considerations for the INTO and UNIQUEINTO command options

The INTO option allows you to select which input records to load into a specific table or partition.

UNIQUEINTO YES tells LOADPLUS that as soon as an INTO specification selects an input record, LOADPLUS should not test the record against other INTO specifications.

Normally, LOADPLUS checks each input record to determine if it matches the WHEN or PART criteria of each INTO specification. Checking starts with the first INTO statement and ends with the last INTO statement. This can be extremely time-consuming when you specify many INTO statements.

INTO and PART

Use the following guidelines to determine how many INTO statements to use with your job and when to use the PART option for optimal performance of your job:

- Avoid specifying a separate INTO statement with PART option for each partition.
- When either of the following conditions exists, specify only one INTO statement without the PART option. The resulting partitioning key determines the partition into which the record is loaded, so the PART option is not necessary.
  - If you specify LOAD REPLACE
  - If you specify LOAD RESUME YES and are loading the majority of partitions in a partitioned table space
- If you specify LOAD RESUME YES and are loading only a few partitions, specify all partitions on one INTO statement (for example, INTO PART 1,3:5). This specification eliminates unloading and loading all partitions of partitioned indexes and can save considerable time.

UNIQUEINTO YES

Use UNIQUEINTO YES if both of the following conditions exist:

- You are loading multiple tables of a nonpartitioned table space or multiple partitions of a partitioned table space.
Each input record matches at most one INTO specification.

As soon as an input record matches the WHEN selection criteria of an INTO statement, LOADPLUS does not check any more INTO specifications. For each input record, this reduces the number of INTO specifications that LOADPLUS checks to half of the INTO specifications made, on average, which can improve performance significantly. You can also improve performance by specifying the INTO statement that selects the most input records, followed by the INTO statement that selects the second most input records, and so on.

Performance considerations for the MAXP installation option

The MAXP installation option limits the number of partitions that LOADPLUS accesses concurrently.

This option permits LOADPLUS to balance parallel processing and use of virtual storage. If the MAXP value is too small, many I/O waits can slow load processing. If the value is too large, excessive paging activity can slow load processing. BMC recommends that you start with a value of 5, which is the value that is shipped with LOADPLUS.

Performance considerations for the ORDER command option

The ORDER command option controls whether LOADPLUS sorts the input data.

In determining how to optimize the performance of LOADPLUS, you must balance your need for fast execution and data availability with your need to have data sorted when it is loaded. The following discussion can help you determine how to specify the ORDER command option to meet your specific performance needs.

ORDER NO

ORDER NO tells LOADPLUS not to sort the data, thus decreasing processing time. However, if your tables have clustering or partitioning indexes and the input data is not in clustering or partitioning key order, you might need to reorganize the table space after the load job to obtain adequate DB2 performance.

In some cases, however, you can improve the performance of your job by specifying ORDER YES.

ORDER YES

Specifying ORDER YES causes LOADPLUS to sort the input data during the data process of the PRELOAD phase or the COMBINED phase. Sorting at this point can
increase the elapsed time of the load process significantly. If you specify ORDER YES, LOADPLUS performs no other processing while the data is being sorted, because no index information can be passed to the index process until BMCSORT starts outputting ordered records. Although ORDER YES increases processing time, the total elapsed time that LOADPLUS requires to sort and load the data is still less than the time required to presort the data and then run LOADPLUS.

However, if you are loading a partitioned table space (except partition-by-growth), you can improve the performance of your load job by specifying ORDER YES. Specifying ORDER YES in this scenario enables multitasking for both loading the data and building the clustering index.

Another advantage of specifying the ORDER YES option is that no data-sorting index work information is required in the SYSUT1 data set, thus reducing I/O operation and DASD space for that data set. Instead, LOADPLUS takes the index information from the data rows during the LOAD phase.

**ORDER PRESORTED**

ORDER PRESORTED tells LOADPLUS that the data is already in the same sequence as the clustering index. Therefore, LOADPLUS does not have to sort the data or the clustering index, saving a significant amount of time. LOADPLUS does verify that the data is in the correct order and terminates the job if it is not.

*Note*

If the data is partitioned and the clustering index is either not partitioned or is a DPSI, the data must be sorted first by partition, then by clustering key within each partition.

**ORDER options with RESUME YES SHRLEVEL NONE**

If you specify LOAD RESUME YES SHRLEVEL NONE, LOADPLUS stops the table space and associated indexes at the beginning of the PRELOAD phase or COMBINED phase. If you must order your data, you need to maximize data availability, and you can afford to lengthen the overall elapsed time to complete the job, presort the data before running LOADPLUS. Specify ORDER NO if you have already sorted the data. When you specify ORDER NO, LOADPLUS does not load any indexes with the data, and avoids the overhead of sorting the data.

**Performance considerations for the PRELOAD ANALYZE command option**

The PRELOAD ANALYZE command option tells LOADPLUS to stop after the optimization phase of PRELOAD.
LOADPLUS displays a message that indicates the optimal number of concurrently processed sort and read tasks. You can use this information to adjust the number of SORTOUT data sets in your JCL.

Performance considerations for the PRELOAD LOAD installation and command options

When you specify LOAD on the PRELOAD installation or command option, LOADPLUS combines the PRELOAD and LOAD processing phases into a single processing phase (the COMBINED phase).

How PRELOAD LOAD improves performance

Because most single-phase load jobs do not use the SORTOUT and SYSUT1 data sets, these types of load jobs provide a significant resource savings when processing large amounts of data, including the elimination of all EXCP processing associated with these data sets. In environments that require the use of tape for the SORTOUT and SYSUT1 data sets, this type of load job can also provide significant CPU and elapsed time savings. Tape mount time becomes nonexistent and CPU consumption can be reduced because there is no read/write activity to these data sets.

Note
For certain types of single-phase jobs, you cannot restart your load job if you do not specify SORTOUT and SYSUT1 data sets.

The fastest load is a single-phase load with LOAD REPLACE ORDER PRESORTED and no participating nonpartitioned indexes. You can specify ORDER PRESORTED on a single-phase load with RESUME YES PART n REPLACE if all of the following conditions apply to your load:

- You are replacing all partitions that are involved in the load.
- No nonpartitioned indexes are participating in the load.
- You use only one SYSREC data set.

Determining whether to specify PRELOAD LOAD

Consider the following information when determining whether to specify PRELOAD LOAD (single-phase load) or PRELOAD CONTINUE (two-phase load):

- Use two-phase processing if you are running jobs that might have a significant number of duplicate keys. In the COMBINED phase, LOADPLUS loads data into the table before checking for duplicate keys in the unique non-data-sorting indexes. At the end of the COMBINED phase, LOADPLUS deletes the rows that cause the duplicates and the duplicate key entries for all indexes for those rows.
— For ORDER YES cases, use two-phase processing only if there are unique non-data-sorting indexes. (Data-sorting index key violations under ORDER YES do not cause performance degradation.)

— For ORDER NO cases, use two-phase processing in this case regardless of the type of participating indexes.

The COMBINED phase of LOADPLUS is a combination of the PRELOAD and LOAD phases. If you specify SHRLEVEL NONE, the table space that you are loading is stopped at the beginning of the phase; therefore, the amount of time that the objects are unavailable might be longer with a single-phase load.

**Performance considerations for the RESUME and REPLACE command options**

The RESUME and REPLACE command options control how LOADPLUS handles table spaces that contain existing data.

Depending on the options that you choose, LOADPLUS might perform an initial analysis of the table space that you are loading, requiring additional processing overhead.

**RESUME YES**

If you specify the RESUME YES option, LOADPLUS must find the location where new data is to be loaded. LOADPLUS reads the table space, looking for the first available page after all existing data. Although this operation is not very time-consuming, it does require that LOADPLUS allocate, open, and perform some I/O operations on the table space data set.

LOADPLUS checks for active data by sequentially reading the table space and searching each page for active rows. If the table space is newly defined, this checking is very fast and requires only one I/O operation on the table space. However, if the table space contains many deleted rows or dropped tables, LOADPLUS must read each page, requiring many I/O operations and slowing processing.

The RESUME YES INDEX UPDATE option of the LOAD command tells LOADPLUS to update indexes rather than unload and reload them. LOADPLUS updates indexes during the LOAD or COMBINED phase. For best performance, when you are adding a small number of rows to an existing table that contains a large number of rows, use the INDEX UPDATE option instead of INDEX BUILD (which is the default).

**RESUME NO**

RESUME NO tells LOADPLUS to load new data at the beginning of the table space. Because this destroys any existing data, LOADPLUS checks for existing data before performing the load operation. LOADPLUS terminates if active rows exist.
REPLACE

REPLACE tells LOADPLUS to ignore any existing data in the table space, so LOADPLUS does not check for active rows. For best performance, when you refresh a table space, use the REPLACE option instead of deleting rows or dropping tables.

Performance considerations for the RULES installation option

Because LOADPLUS does not convert data to internal format when the installation option RULES=STANDARD is in effect, WHEN, DEFAULTIF, and NULLIF comparisons that LOADPLUS performs under RULES=STANDARD always outperform those that LOADPLUS performs under RULES=BMC.

Note

The RULES option interacts with the FORMAT option on the LOAD command as follows:

- When you specify FORMAT UNLOAD, FORMAT BMC, or FORMAT BMCUNLOAD, LOADPLUS changes the value of the RULES option to BMC, regardless of the value that you specified at installation.

- When you specify FORMAT CSV, LOADPLUS changes the value of the RULES option to STANDARD, regardless of the value that you specified at installation.

When RULES=BMC and you specify a field name in your predicate where the field name matches a column name in the table that you are loading, LOADPLUS converts the value that the field defines to the data type and length of the column before it evaluates the predicate. (This is different than the IBM DB2 LOAD utility which only performs a byte-by-byte comparison based on the byte length of the field.) Depending on the conversion that is involved, converting the value can be time-consuming and is always more so than a predicate that uses (start:end) or a field without a corresponding column. Therefore, to achieve the best performance, specify your predicates by using (start:end) or a field name that does not correspond to a column.

Performance considerations for the sort processing options

The BMCSORT technology provides LOADPLUS with more control of the sort process than external sort routines provide.

This added control helps prevent memory-related problems during the sort process. LOADPLUS allocates the amount of resources to each sort process based on the amount of work that LOADPLUS determines that sort process will perform.
The sort optimization routines that LOADPLUS uses normally provide the best performance. However, on rare occasions, you might need to make adjustments to enhance the performance of LOADPLUS and other applications running on your system. The following sections provide information that can help you improve the performance of your sort processing when necessary.

**Controlling memory usage (SMCORE)**

LOADPLUS provides several installation and command options to give you control, when necessary, over the amount of memory that BMCSORT uses during a load job.

The SMCORE installation option contains two parameters: total memory and below-the-line memory. BMC strongly recommends that you use the values 0K and 0K for these parameters. These values generally provide the highest sort performance and tell LOADPLUS to determine the appropriate amount of memory to use for each sort based on the following criteria:

- Amount of data or index keys to be sorted
- Value that you specify for REGION in either your JCL or system exits
- Amount of memory that is available during optimization
- Number of sorts to process
- Minimum and maximum amounts of memory indicated by the MINSORTMEMORY and MAXSORTMEMORY installation or command options
- Percentage of available or total 4-kilobyte system pages that you specify with the AVAILPAGEPCT and TOTALPAGEPCT options

**Note**

The SHORTMEMORY installation and command options control the action that LOADPLUS takes when a memory shortage exists.

**Total memory**

The first parameter value of the SMCORE option tells LOADPLUS how much total memory, both above and below the 16-megabyte line, that you want BMCSORT to use during a single invocation. BMC strongly recommends that you specify 0K, which allows LOADPLUS to determine the optimal amount of total memory to use. However, other valid values are 4096K through 65536K. You can also specify this value in megabytes (either 0M or 4M through 64M).

Whether LOADPLUS determines the value for total memory or you specify a value, LOADPLUS multiplies this value by the number of required sort processes to determine a value for the total memory that the current job requires. Depending on
the workload and system environment, LOADPLUS distributes this total memory among the sort processes for the job.

For example, if you specify 4096K for the total memory parameter and LOADPLUS determines that it needs four sort processes for this job, LOADPLUS calculates that it needs 16384 KB of total memory for the job. However, if the workload for each sort process is different, LOADPLUS invokes BMCSORT for each sort process with varying amounts of memory. Some of these amounts will be lower or higher than the 4096 KB that you specified.

The following additional considerations apply to the first SMCORE value:

- The region size that is available for your load job in conjunction with the value that you specify for this parameter can constrain the number of sort processes that LOADPLUS starts. Because the region size must include space for buffers and other required structures, the entire region size is not available for sort processing. You can avoid this constraint by using a value of 0K, allowing LOADPLUS to determine the optimal amount of total memory to use.

- When you allow LOADPLUS to optimize total memory, LOADPLUS never uses more memory than the value of your region parameter.

- If you specify a value other than 0 for this parameter, LOADPLUS ignores the MINSORTMEMORY, MAXSORTMEMORY, and SHORTMEMORY values.

**Below-the-line memory**

The second parameter value of the SMCORE option indicates how much memory BMCSORT should use below the 16-megabyte line during a single invocation. BMC recommends that you specify 0K, which allows LOADPLUS to determine the optimal amount of below-the-line memory to use. In addition to 0K, valid values are 256K through 4096K. You can also specify this value in megabytes (either 0M or 1M through 4M).

Specifying a value greater than 384K can limit the number of sort tasks that LOADPLUS can start concurrently.

**Controlling the number of sort processes (SMAX or MAXSORTS)**

LOADPLUS determines the optimal number of sort processes to execute concurrently based on available resources. LOADPLUS calculates the optimal number of concurrent sort processes based on the following values:

- The number of SORTOUT data sets that you specify
- The number of sort work data sets that you specify
- The number of sort processes that will fit in available memory
The number of partitions that are participating in the LOADPLUS job

- The value of the SMAX installation option or MAXSORTS command option
- The value of the MAXP installation option

You can specify the maximum number of concurrent index sort processes by using the SMAX installation option or, for LOAD-phase index sort tasks, the MAXSORTS command option. BMC recommends a value of 16 for SMAX. However, if system resources are constrained or other problems arise, you can change SMAX or MAXSORTS to limit the number of concurrent sort processes.

**Performance considerations for the TSSAMPLEPCT installation and command option**

The TSSAMPLEPCT option controls table space sampling during statistics gathering.

Using a larger value for this option might impact job performance but provide more accurate results. Balance these considerations when choosing a value. For example, by improving the accuracy of the DB2 catalog statistics that you update, you provide the DB2 optimizer with better access paths, which can improve the performance of your applications.

**Performance considerations for the UXSTATE installation option**

LOADPLUS can invoke the following types of DB2 user exits during processing:

- EDITPROCs
- VALIDPROCs
- FIELDPROCs
- Date exits
  - DSNXVDTX (EBCDIC date exit routine)
  - DSNXVDTA (ASCII date exit routine)
  - DSNXVDTU (Unicode date exit routine)
- Time exits
  - DSNXVTMX (EBCDIC time exit routine)
  - DSNXVTMA (ASCII time exit routine)
— DSNXVTMU (Unicode date exit routine)

- An authorization exit

By default, LOADPLUS invokes these exits in supervisor state and PSW key=7. LOADPLUS calls the MODESET SVC to perform this switch before invoking an exit, and again to reset the mode after returning from the exit.

However, invoking MODESET is an expensive CPU process. If you can ensure that all exits that LOADPLUS calls can run in problem state, you can specify UXSTATE=PROB in the installation options. This option causes LOADPLUS to invoke user exits in problem state (PSW key=7) and achieves significant savings in CPU overhead.

**Performance considerations for the XBLKS installation option**

Transfer blocks (XBLKs) perform the following functions in the PRELOAD phase or the COMBINED phase of LOADPLUS:

- Moving data from the read task to the sort/data task

- Moving non-data-sorting index records from the sort/data task to the probe or index task

The read task reads data from SYSREC, completes all conditional processing, and moves the row image into an XBLK. If multiple sort/data tasks exist, an XBLK is associated with each task. When an XBLK is full, it becomes available for processing by the sort/data task.

The sort/data processes pass information to the probe or index task by using one or more XBLKs. The probe task is used when checking referential constraints. The index task is used when an additional sort process is necessary to check for duplicate keys.

The number of XBLKs that LOADPLUS allocates is a function of the number of read, sort/data probe, and index tasks that are active for a particular LOADPLUS job. The XBLKs installation option specifies how many XBLKs to allocate per receiving task. The minimum value for the XBLKs option is 3, which is the value that is shipped with LOADPLUS. A larger number can marginally improve performance, but can also increase paging. XBLKs are 64 KB each and are allocated from storage below the bar.
Enabling multitasking for performance in LOADPLUS

During execution, LOADPLUS determines the most effective arrangement of tasks when running in a multi-processor environment.

Although LOADPLUS runs very efficiently on single-processor computers, it performs most efficiently on large multi-processor systems.

Loading data into DB2 tables requires several distinct tasks, including converting data, verifying indexes, loading data, building indexes, and creating image copies. LOADPLUS can perform these tasks concurrently, thus reducing the elapsed time of the load process. For example, if partitioned clustering or partitioning indexes exist and you specify the ORDER YES option, the load process builds these indexes while it is loading the data. The key values from these indexes are taken from the rows as they are loaded into the table space. Index pages are built and written as they fill up. This process provides more overlapped I/O processing. As an example, a partitioned table space that is loaded with ORDER YES and a MAXP value of 5 will actually have 10 data sets open concurrently.

You can enable additional multitasking by specifying multiple data sets for many of the data set types that LOADPLUS uses, including input data sets, work data sets, and copy data sets. The following sections discuss additional performance considerations.

Using multiple SYSREC data sets

LOADPLUS multitasks as much as possible in all cases.

When loading partitioned table spaces (except partition-by-growth), you can obtain additional multitasking benefits when multiple SYSREC data sets are available. For restrictions on multiple SYSREC data sets, see “SYSREC data sets in LOADPLUS” on page 456.

Multiple SYSREC support provides the maximum throughput in the least amount of elapsed time when loading partitioned table spaces. While all types of load processes support multiple SYSREC data sets, LOADPLUS reads multiple SYSREC data sets concurrently only when loading a partitioned table space (except partition-by-growth). LOADPLUS allows up to 256 SYSREC data sets and processes up to 16 SYSREC data sets concurrently.

When using multiple SYSREC data sets, you can gain the greatest performance improvement during large load jobs when there is a relatively even distribution of rows across partitions. Note the following performance recommendations when specifying multiple SYSREC data sets:

- For a two-phase load, the number of SYSREC data sets should be about the same as the number of SORTOUT data sets or should be an even multiple of the number of SORTOUT data sets.
If you have more than 16 input data sets of different sizes, BMC recommends that you arrange them from the largest to the smallest.

These data sets are not partition dependent. Data for a particular partition can be in one or more SYSREC data sets. There is no advantage to putting all of the data for one partition into one SYSREC data set.

Using multiple SORTOUT data sets

For optimum performance, BMC recommends that you use single-phase processing, which does not require SORTOUT data sets.

However, if you must use two-phase processing, this section describes performance implications and recommendations when using multiple SORTOUT data sets.

LOADPLUS allows you to specify up to 16 SORTOUT data sets. However, the number of SORTOUT data sets that LOADPLUS processes concurrently might be smaller than the number that you specify.

LOADPLUS uses what it determines to be the optimal number of concurrent sort processes. The number of SORTOUT data sets that you specify limits the number of concurrent sort processes and can also affect how LOADPLUS determines the number of optimal sort processes. There is no advantage to specifying more data sets than the optimizer will select for use. See “Performance considerations for the sort processing options” on page 665 for more information about how LOADPLUS determines the number of sort processes to use.

Specifying PRELOAD ANALYZE tells LOADPLUS to stop after optimization and display a message that provides the optimal number of concurrent sort and read tasks. You can use this information to adjust the number of SORTOUT data sets in your JCL.

Note the following performance information when using multiple SORTOUT data sets with partitioned table spaces:

- When loading partition-by-growth table spaces, LOADPLUS uses only one SORTOUT data set, regardless of the number that you specify.

- If a single partition is to receive 40% or more of the data, you should not use multiple SORTOUT data sets.

- The number of SORTOUT data sets should be an even factor of the number of partitions. For example, a table space with 6 partitions should use either 2, 3, or 6 SORTOUT data sets.

Using multiple SYSUT1 data sets

Using multiple SYSUT1 data sets provides the following performance advantages:
- I/O processing to each SYSUT1 data set is overlapped with other I/O processing and with CPU processing.

- LOADPLUS writes any non-unique index information when the index process receives it and does not pass the information to BMCSORT, thus reducing the amount of data that is sorted. This process also reduces the amount of DASD space that is required for all index information.

- With a single SYSUT1 data set, LOADPLUS pads all keys to the length of the longest key that is being processed. Padded keys require more DASD space and more I/O operations are required to process the index information. With multiple SYSUT1 data sets, no padding of keys is needed.

- If LOADPLUS finds that enough resources (that is, virtual storage and sort work space) are available, LOADPLUS starts multiple tasks to build the indexes concurrently in the LOAD phase.

If you are using multiple index work files, specify at least one SYSUT1 data set for each participating non-data-sorting index. If the number of SYSUT1 DD statements is greater than one and less than the number of participating non-data-sorting indexes, LOADPLUS terminates with an error. Also, when ORDER NO is in effect, none of the indexes are designated as data-sorting indexes. Therefore, if you specify ORDER NO, you must specify one SYSUT1 data set for each participating index.

**Note**

If you are loading a table space with a large number of indexes, BMC recommends that you specify a single SYSUT1 data set to avoid data set allocation limitations of the operating system.

When you specify multiple SYSUT1 data sets, the DD statement specification is SYSUT1nn, where nn is a unique suffix for each DD statement. The suffix is not used to identify which index is assigned to a specific SYSUT1 data set.

### Additional performance tuning recommendations for LOADPLUS

This topic describes additional steps that you can take to improve the performance of your load jobs.

**Tuning I/O**

Because LOADPLUS reads and writes large amounts of data during typical load processing, one of the most important factors that affects LOADPLUS performance is I/O processing. To maximize I/O performance, LOADPLUS handles all of its own buffering, and performs I/O operations at the lowest level possible. This allows
reading or writing of several blocks of data with each I/O operation and allows
LOADPLUS to prefetch subsequent data.

In addition to the information in “Performance considerations for buffer options” on page 656 and “Performance considerations for the sort processing options” on page 665, the following information can help you tune your I/O processing:

■ Allocate input data sets with the largest block size possible to reduce the number of I/O operations that LOADPLUS requires to read the data set. LOADPLUS always calculates the optimum block size for output data sets.

■ To avoid I/O queueing, allocate LOADPLUS data sets on separate channels and drives. If you do not have sufficient channels available, use separate drives and control units.

Providing maximum virtual storage

Because each task requires virtual storage for processing, LOADPLUS balances the multi-processing that it performs with the amount of virtual storage that it finds available. The primary use of virtual storage is for I/O buffers by either LOADPLUS or BMCSORT.

Because LOADPLUS uses as much virtual storage as needed, always specify as much virtual storage as you can. BMC recommends that you specify REGION=0M in your JOB or EXEC statement to tell the system to allocate all available virtual storage to the LOADPLUS job. If your data center does not permit you to specify REGION=0M, specify the amount that allows the most virtual storage both above and below the 16-megabyte line.

Using a region size that is less than optimal risks the following potential issues:

■ Running less efficiently, which could result in additional CPU and elapsed time

■ Encountering memory failures or jobs that fail when new versions implement changes that require additional memory

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 or EXEC statement. BMC makes the following recommendations for the MEMLIMIT option:

■ Specify NOLIMIT to allow unlimited above-the-bar memory.

■ If you are unable to specify NOLIMIT, specify at least 4 GB; if you are loading LOB or XML data, specify at least 32 GB.
Performance tuning for specific scenarios in LOADPLUS

The following sections describe steps that you can take to tune specific types of load jobs to improve performance. Use the following table to locate a specific load scenario:

Table 116: Performance tuning scenarios

| Scenario type       | Scenario                                      | Reference                                                      |
|---------------------|-----------------------------------------------|                                                               |
| Input data          | Data from UNLOAD PLUS                         | “Loading data from UNLOAD PLUS (FORMAT BMCUNLOAD option)” on page 675 |
|                     | XML data                                       | “Loading XML data” on page 675                                |
|                     | LOB data                                       | “Loading LOB data” on page 675                                |
|                     | Multiple input files                           | “Loading data from multiple input files” on page 675          |
|                     | Single input file                              | “Loading data from a single input file” on page 675           |
| Object              | Tables with DEFINE NO data sets                | “Loading tables with DEFINE NO data sets” on page 676         |
|                     | Tables with identity columns                   | “Loading tables that contain identity columns” on page 676    |
|                     | Partitioned table spaces                       | “Loading partitioned table spaces” on page 676                |
|                     | Objects with a large number of partitions      | “Loading objects with a large number of partitions” on page 677 |
|                     | Databases with many data sets and tables       | “Databases with many data sets and many tables” on page 677    |
| High availability   | SQLAPPLY load                                  | “SQLAPPLY load” on page 678                                   |
|                     | SHRLEVEL CHANGE (except an SQLAPPLY load) or SHRLEVEL REFERENCE | “SHRLEVEL CHANGE (except an SQLAPPLY load) and SHRLEVEL REFERENCE” on page 678 |
| Additional features | Sorted data                                    | “Sorted data” on page 679                                     |
|                     | Copies                                         | “Copies” on page 679                                          |
|                     | Statistics                                     | “Statistics” on page 679                                      |
Performance tuning by type of input data

If you are loading data in certain environments, the type of input that you provide can affect the performance of your load job.

Loading data from UNLOAD PLUS (FORMAT BMCUNLOAD option)

When used in conjunction with UNLOAD PLUS, LOADPLUS provides a high-speed option to move data from one table to another table that has a similar table structure. This feature is useful, for example, for migrating data from test to development databases. To unload the data, you specify the FORMAT BMLOAD option in UNLOAD PLUS. UNLOAD PLUS unloads the data in an internal format that only LOADPLUS can use. To load the data, you specify the FORMAT BMCUNLOAD option on your LOAD command.

Because the data is never converted to an external format and because LOADPLUS does not perform data verification, using this option can help reduce CPU cycles and elapsed time for your load job as well as for the entire process of migrating data.

Loading XML data

When loading XML data, adjust the XMLAVGSIZE global option or AVGSIZE field option to improve the performance of your load job.

Loading LOB data

When loading LOB data, adjust the LOBAVGPC global option or AVGSIZE field option to improve the performance of your load job.

Loading data from a single input file

When loading data from a single input file and the data is already sorted, specify ORDER PRESORTED to obtain optimal performance.

Loading data from multiple input files

When you are loading data from multiple input files, use multiple SYSREC data sets. LOADPLUS can run three read tasks per active CPU in an LPAR. The optimal configuration is to distribute the amount of data evenly across all of the SYSREC DDs. In addition, you can concatenate multiple data sets to a single SYSREC. Use the following guidelines for setting up specific types of load jobs:

- If one file has half of all of the data and the rest of the data is in more than one additional file, allocate the one large file to one SYSREC, and the rest of the files concatenated to a second SYSREC. The input process is never shorter than the
longest task, so adding additional SYSREC files does not decrease the elapsed
time and uses additional resources that could be allocated more productively.

- If you have 16 data sets and want to run four input tasks, specify SYSREC01
  through SYSREC04 and have four data sets allocated to each SYSREC nn.

For requirements when using multiple SYSREC data sets, see “SYSREC data sets in
LOADPLUS” on page 456.

**Performance tuning by type of object**

Use the following information to tune your jobs when loading specific types of
objects.

**Loading tables with DEFINE NO data sets**

As required by DB2, if you are loading at least one partition of a table whose table
space or index spaces are defined with DEFINE NO, LOADPLUS materializes all
partitions. For a table space with many partitions, this materialization might affect
the performance of your load job. Subsequent jobs on the same table space, however,
are not affected.

**Note**

For LOB table spaces, LOADPLUS materializes the LOB spaces only for base table
partitions that are participating in the load.

**Loading tables that contain identity columns**

If you are loading an identity column during an SQLAPPLY load job, LOADPLUS
does not generate values but allows DB2 to assign values during insert processing.
Loading an identity column by specifying LOAD RESUME YES SHRLEVEL
CHANGE SQLAPPLY can reduce the performance of your load job.

**Loading partitioned table spaces**

Consider the following performance recommendations if you are loading partitioned
table spaces (except partition-by-growth). If the objects have a large number of
partitions, see also the recommendations in “Loading objects with a large number of
partitions” on page 677.

- Use multiple SYSREC data sets (and, if you must use two-phase load, multiple
  SORTOUT data sets).

- If you are loading a majority of the partitions, specify a single INTO statement
  without the PART option. If you are loading a small number of partitions, specify
  all partitions on a single INTO statement (for example, INTO PART 1:3,5).
■ Specify PART for the COPYLVL installation or command option if you need to make copies.

■ Consider specifying ORDER YES to allow LOADPLUS to more efficiently build the data-sorting indexes and load the data.

**Loading objects with a large number of partitions**

If you are loading objects with a large number of partitions, consider the following performance recommendations. For more considerations, including a definition of large number of partitions, see “Considerations when loading a large number of partitions” on page 105.

■ If you are replacing only a few partitions and the other partitions are empty, specify LOAD RESUME YES PART \texttt{n} REPLACE instead of LOAD REPLACE.

■ If you are loading a large number of partitions, consider specifying REDEFINE NO, either in the installation options or on the LOAD command. This option creates additional data sets when needed and cleans up unused data sets.

■ If you are loading a large number of partitions, need to make inline copies, and are specifying ORDER NO, consider specifying COPYLVL FULL.

■ If you are loading a large number of partitions, ensure that you specify REGION=0M in the JOB or EXEC statement of the execution JCL to tell the system to allocate all available virtual storage to the LOADPLUS job.

**Databases with many data sets and many tables**

LOADPLUS offers features that provide advantages over competitors when loading these types of databases. For example, LOADPLUS allows you to load data into tables whose table spaces and index spaces are defined with DEFINE NO.

If you are using LOADPLUS in this type of environment, consider the following performance recommendations:

■ Consider specifying REDEFINE NO, either in the installation options or on the LOAD command. This option creates additional data sets when needed and cleans up unused data sets.

■ When replacing data in a segmented table space, specify LOAD RESUME YES TABLE tableName REPLACE to load only the tables in which you want to replace data.

■ Consider skipping processing on nonpartitioned indexes by using the SKIPIX option.
Performance tuning and high availability

If you need the best performance while maintaining the highest availability of your data, you can use the online load features of LOADPLUS.

SQLAPPLY load

Specify LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY or LOAD RESUME YES SHRLEVEL CHANGE without PART n REPLACE to allow your objects to remain in RW status during the load process.

To obtain optimal performance with this option, note the following information:

- If there are low cardinality keys and there is high CPU usage, the values for APCOMMIT and APMAXAGENTS can impact performance. Adjust one parameter at a time to determine the best combination of values.

  Note
  APMAXAGENTS affects only partitioned or multi-table table spaces.
  LOADPLUS uses up to one agent per partition or table. For partition-by-growth table spaces, LOADPLUS uses only one agent per load regardless of the value that you specify for APMAXAGENTS.

- If you are loading tables that contain identity columns, running an SQLAPPLY load can degrade performance.

SHRLEVEL CHANGE (except an SQLAPPLY load) and SHRLEVEL REFERENCE

SHRLEVEL REFERENCE and SHRLEVEL CHANGE allow your objects to remain in RO or RW status during the load process. For SHRLEVEL CHANGE (except an SQLAPPLY load) and SHRLEVEL REFERENCE, LOADPLUS uses staging data sets. During the UTILTERM phase, LOADPLUS either renames the staging data sets or uses FASTSWITCH processing to have the DB2 catalog point to the staging data sets. For optimal performance, consider the following options:

- Specify FASTSWITCH YES if possible.

- If you choose the rename process (FASTSWITCH NO), predefine the staging data sets and specify REDEFINE NO. This option creates additional data sets when needed and cleans up unused data sets.

Performance tuning by additional feature

Use the information in this topic to tune your load jobs if you need specific results, such as sorted data or updated statistics.
Sorted data

To obtain optimal performance when you need sorted data, consider the following information:

- If you are using UNLOAD PLUS to unload and LOADPLUS to load data, and you need sorted data, perform the sort in UNLOAD PLUS rather than LOADPLUS.

- If you are loading from a single input file and the data is already sorted, specify ORDER PRESORTED.

For more information about optimal performance with sort processing, see “Performance considerations for the sort processing options” on page 665.

Copies

Consider specifying PART for the COPYLVL installation or command option if you are loading a partitioned object. Specifying COPYLVL PART tells LOADPLUS to make copies by partition. However, if you are loading a large number of partitions, consider specifying COPYLVL FULL or limiting the number of copies per partition to avoid data set allocation restrictions of the operating system or LOADPLUS memory restrictions.

Statistics

If you need updated statistics, you can obtain optimal performance by specifying UPDATEDB2STATS YES on the LOAD command instead of running RUNSTATS after the load job. Ensure that you have run RUNSTATS on this object at least once prior to this load job.

Interpreting performance-related messages from LOADPLUS

LOADPLUS issues performance-related messages if you specify MSGLEVEL(1) on your EXEC statement or in your installation options.

Information that these messages provide can help you monitor the performance of LOADPLUS and fine-tune future runs. For message explanations, access the BMC Documentation Center from the BMC Support Central site (http://www.bmc.com/support).

**BMC50364I** SORT PROCESSES SEVERELY CONSTRAINED BY MEMORY RESOURCES. SORTING CONTINUES, BUT PERFORMANCE MAY BE IMPACTED

This message indicates that the utility found that the memory resources available are insufficient to provide the most efficient sorting. To allow the utility job to complete,
the sort routine might choose a less efficient sorting algorithm, which can increase the elapsed time of the job.

Also see message BMC50474I, which reports the memory that is available to the utility at the time of optimization. If the value for ABOVE as indicated in BMC50474I is substantially less than the value for your REGION parameter, your system might have a memory-limiting exit active.

One of the following actions might improve the sort efficiency for future jobs:

- Increase your region size, if possible.
- Limit the scope of the job that you are running to reduce the amount of work that the utility needs to do.
- If your system has a memory-limiting exit active, contact your systems programmer to increase the memory limit, if possible.

**BMC50397I** phase PROCESSING CONSTRAINED BY REGION SIZE

This message indicates that the load job is constrained because the amount of virtual memory available is insufficient for optimal performance. For most jobs, LOADPLUS continues processing. If you also receive message BMC50399E, however, the amount of virtual memory available to LOADPLUS is insufficient to continue.

BMC recommends that you specify REGION=0M in the JOB or EXEC statement of the execution JCL to tell the system to allocate the optimal amount of available virtual storage to the LOADPLUS job. However, if you are processing a large number of partitions, processing might be constrained, even if you specified REGION=0M. In this case, consider loading fewer partitions in a single job.

**BMC50398I** phase PROCESSING CONSTRAINED BY SORT WORK FILES

This message indicates that the load job is constrained because the number of sort work files or the total amount of space for the sort work files is insufficient for optimal performance. For most jobs, LOADPLUS continues processing. If you also receive message BMC50399E, however, the number or size of the sort work files is insufficient for LOADPLUS to continue.

**BMC50399E** phase PROCESSING UNABLE TO CONTINUE DUE TO CONSTRAINED RESOURCES

This message indicates that the load job cannot get enough memory or sort work space to process the indicated phase. Message BMC50397I, message BMC50398I, or both are issued before this message.

Make one or more of the following adjustments:

- Specify a larger region size.
Specify larger sort work files or more sort work files.

Adjust the installation options to improve performance.

If this message is accompanied by message BMC50397I but you have specified REGION=0M in your JOB or EXEC statement, processing might be constrained because you are loading a large number of partitions. If this is the case, try loading fewer partitions in a single job.

**BMC50400I** phase PROCESSING CONSTRAINED BY INDEX WORK FILES

This message indicates that the load job is constrained because there are not enough index work files, but the job continues.

**BMC50471I** environmentInformation

This message displays current values for each option in the installation options module. Use this information to verify that LOADPLUS is using the option value that you want to use.

**BMC50474I** BELOW 16M = nK, ABOVE 16M = nK, CPUS = n

This message provides information about virtual storage and CPU usage. The message displays the following information:

- Amount of virtual storage available below the 16-megabyte line
- Amount of virtual storage available above the 16-megabyte line
- Number of physical CPUs available in the processor

Use this information to ensure that adequate virtual storage is available for LOADPLUS to use. See “Providing maximum virtual storage” on page 673. For more information, see messages BMC50475I and BMC50479I.

**BMC50476I** DDNAME = ddname, I/OS = n, I/O WAITS = w, RDB LOCK WAITS = r

This message provides performance information about I/O operations to sequential data sets. The message displays the following information:

- ddname associated with the I/O operations
- Number of I/Os (blocks) written to or read from the data set
- Number of waits issued for I/O completion
- Number of waits for serialization of the data set

A wait count that is greater than 10 percent of the block count might indicate degraded performance. Try allowing LOADPLUS additional buffer space for the associated data set (see “Performance considerations for buffer options” on page...
or using multiple data sets. A high number of serialization waits might indicate the need to decrease the number of concurrent tasks or use multiple data sets.

This message provides performance information about I/O operations to VSAM data sets. The message displays the following information:

- Processing task number
- Table space partition number
- Number of rows or keys in the partition
- Number of waits issued for I/O completion
- ddname associated with the I/O operations

Use the ddname to find the actual number of I/O processes that were issued to the associated data set. A wait count that is greater than 20 percent of the actual I/O processes might indicate degraded performance. Try allowing LOADPLUS more virtual storage if LOADPLUS issues message BMC50397I. Provide more sort work file space if LOADPLUS issues message BMC50398I. Changing these values enables LOADPLUS to start more concurrent tasks.

This message displays information that LOADPLUS obtains from the operating system (not the region). The allowedAvailablePages value is the value that you specified for the TOTALPAGEPCT option, and allowedFreePages is the value that you specified for the AVAILPAGEPCT option.

Available pages are defined as pages that are underutilized and are available for use. Free pages are defined as pages that have not been used. Because available pages are subject to more system paging than free pages, available pages are more likely to affect system performance.

LOADPLUS considers the percentage of available and free pages when determining the maximum amount of memory that is allowed for sort processing. If insufficient memory exists to perform a minimum number of optimal sort processes, the job terminates with a constrained memory message.

LOADPLUS for DB2 Reference Manual
This message displays the amount of memory below the line, the maximum amount of memory, and the number of 4-KB pages of hyperspace that the utility allows for each sort task.

BMC51495I phase OPTIMIZATION, RC = returnCode, #SORTS = n, #READERS = r, INDEX TASKS = i, TIME = time

This message provides performance information about optimizing the number of sort processes during the PRELOAD phase. Each iteration of the optimizer produces a message and supplies the following information:

- Return code
- Number of sort processes that are being tested for this iteration of the optimizer
- Number of input data sets (SYSREC data sets) that LOADPLUS will read concurrently if this is the optimal number of sort processes
- Number of index tasks that the load job will run concurrently
- *Relative* amount of time that this configuration of read and sort processes will take to process the data

LOADPLUS uses the return code to monitor the success of the optimization. If at least one return code in the series of messages is 0, the PRELOAD optimization completed successfully. When optimization completes successfully, LOADPLUS follows this message with BMC51496I, which provides the optimal number of sort processes.

If no return code is 0, LOADPLUS follows this message with BMC51497E, indicating that optimization failed. You might be able to correct the problem by increasing the REGION parameter or by specifying REGION=0M and resubmitting the job.

For more information about how the optimizer determines the optimal number of sort processes, see “Controlling memory usage (SMCORE)” on page 666.

BMC51499I phase OPTIMIZATION, RC = returnCode, #SORTS = n, #READERS = r, #INDEX TASKS = i, #PROBE TASKS = p, #LOB TASKS = l, #XML TASKS = x, TIME = time

This message provides performance information about optimizing the number of sort processes during the COMBINED phase. Each iteration of the optimizer produces a message and supplies the following information:

- Return code
- Number of sort processes that are being tested for this iteration of the optimizer
- Number of input data sets (SYSREC data sets) that LOADPLUS will read concurrently if this is the optimal number of sort processes
• Number of index tasks that the load job will run concurrently
• Number of probe tasks that the load job will run concurrently
• Number of LOB tasks that the load job will run concurrently
• Number of XML tasks that the load job will run concurrently
• *Relative* amount of time that this configuration of read and sort processes will take to process the data

LOADPLUS uses the return code to monitor the success of the optimization. If at least one return code in the series of messages is 0, the optimization completed successfully. When optimization completes successfully, LOADPLUS follows this message with BMC51500I, which provides the optimal number of sort processes.

If no return code is 0, LOADPLUS follows this message with BMC51497E, indicating that optimization failed. You might be able to correct the problem by increasing the REGION parameter or by specifying REGION=0M and resubmitting the job.

For more information about how the optimizer determines the optimal number of sort processes, see “Controlling memory usage (SMCORE)” on page 666.

```
BMC51507I transferType = b, XFERS = t, EMPTY WAITS = w, FULL WAITS = f
```

This message provides PRELOAD phase or COMBINED phase performance information about data transfer between these tasks:

• The read task and the sort/data task
• The sort/data task and the probe, index, or SQLAPPLY task

This message provides the following information, summarizing the information that BMC51510I provides:

• Number of XBLKs or High-speed Apply buffers available for data transfer
• Number of waits for an empty XBLK or High-speed Apply buffer
• Number of waits for a full XBLK

```
BMC51508I MAX taskType TASKS = t, objectType PER TASK = i, SORTWKS PER TASK = s, MAX OPEN PARTITIONS PER TASK = o
```

This message provides the results of task optimization during build processing. The message displays the following information:

• Maximum number of concurrent tasks
• Maximum number of partitions or indexes that LOADPLUS can process per task
• Number of sort work files that are assigned to each task
• Maximum number of open partitions per task
The maximum number of tasks depends on the system resources that are available, such as virtual storage, the number of physical CPUs, the number of SYSUT1 and SORTOUT data sets, and sort work file space. Increasing the amount of virtual storage, allocating more sort work files, or both might allow LOADPLUS to run more tasks concurrently.

Based on the size of the partition and available sort work file space, LOADPLUS determines the number of partitions to process per task that allows the sort to run in the fastest manner. The actual number of partitions that LOADPLUS processes per task might be less than the maximum that the message shows.

This message provides information about XBLK or High-speed Apply buffer processing for a specific task. The message displays the following information:

- Task number
- Task type (READ, SORT, PROBE, INDEX, SQLAPPLY, DISCARD)
- Number of data transfers during task processing
- Number of waits for an empty XBLK or High-speed Apply buffer
- Number of waits for a full XBLK

For XBLKs, you can use the information that this message provides to refine the number of read and sort/data tasks that LOADPLUS will use. The following table provides information about how to refine these tasks.

**Note**

A read task usually runs faster than a sort/data task when a sort process is involved (ORDER YES). A probe or index task generally runs faster than a sort/data task. When the job does not require a sort of the data (ORDER NO), the sort/data task can be faster than the read task.

There are two pools of XBLKs. LOADPLUS uses one pool for transfers from the read to the sort/data tasks. LOADPLUS uses the other pool for transfers from the sort/data tasks to the probe, index, or SQLAPPLY tasks.
### Table 117: Refining read and sort/data tasks

<table>
<thead>
<tr>
<th>Situation</th>
<th>Indicates</th>
<th>Action</th>
</tr>
</thead>
</table>
| **EMPTY WAIT counts**<br>Read tasks quickly fill all available XBLKs. It is not unusual to see an EMPTY WAIT count that is almost equal to the XFER count. | An EMPTY WAIT count that is much greater than the XFER count might indicate an excessive number of read tasks. This situation is usually caused by an overloaded processor complex, indicating some performance degradation. | Reduce the number of read tasks by using fewer SYSRECnn DDs, if all of the following conditions are true:  
- Performance degradation is indicated.  
- It is the normal time to run the load.  
- The processor load is normal for the time frame. |
| **Excessive FULL WAIT counts from read**<br>Sort/data tasks use both pools of XBLKs. | The FULL WAIT count indicates how often the sort/data tasks waited for data from the read tasks to process. | If the number exceeds 10% of the XFER counts from the read tasks, LOADPLUS can effectively use additional read tasks. If the job has concatenated DDs, provide separate SYSREC nn ddnames for each data set. |
| **Excessive EMPTY WAIT counts and FULL WAIT counts** | | If the sort/data tasks report a high number of EMPTY WAITs and the probe or index task shows a high number of FULL WAITs, consider increasing the number of XBLKs. Weigh the benefits of increasing the number of XBLKs against the potential problems that might occur with memory utilization and paging rates.  
Changing the XBLKS installation option affects all LOADPLUS executions. If you decide to alter the option, proceed slowly, in increments of one per day.  
Observe and evaluate the results before leaving the new setting or increasing it again. |

---

**Tuning to improve memory use in LOADPLUS**

To help you determine how to improve memory usage, consider the information in the following topics.
**Memory requirements of the ORDER options**

Jobs that use ORDER NO require that all partitions of a table space be open at one time and therefore have a higher memory requirement.

Jobs that use ORDER YES or ORDER PRESORTED require that only one partition per task be opened at a time. ORDER PRESORTED also eliminates the need to sort the clustering index.

**Tuning memory use in sort processing**

Specifying a value greater than 0 for the SORTNUM option (with DDTYPE SORTWORK ACTIVE NO and no sort work data sets specified in the JCL) tells LOADPLUS that BMCSORT will control sort work allocations, which should eliminate sort work constraints.

The amount of multitasking that sort processing performs depends on the number of CPUs, the SMAX or MAXSORTS option value, the number of sort work data sets, the number of SORTOUT data sets, and available memory. You might be able to improve memory usage by adjusting the SMAX or MAXSORTS option value or the number of sort work data sets.

**Using multitasking to improve memory use**

Multitasking can improve memory use in your load jobs.

For information, see “Enabling multitasking for performance in LOADPLUS” on page 670.

**Tuning memory use in copy processing**

The number of copies that you make during a load job can affect the memory that your system uses.

Balance your organization’s backup copy needs with your system’s memory use requirements to determine the optimal number of copy data sets to allocate.

Memory constraints and data set allocation constraints are particularly critical when you are making copies while loading a large number of partitions. Consider one of the following options in this case.
If you are dynamically allocating copy data sets, specify COPYLVL FULL to have LOADPLUS allocate a single copy data set for the table space.

If you are loading a subset of contiguous partitions, create a single copy by specifying COPYSUBSET=YES in the installation options module. Also, if you are dynamically allocating copy data sets, specify COPYLVL FULL on your LOAD command.

If you need copies by partition, specify as few copies per partition as possible or consider specifying fewer partitions per job.

**Tuning for availability**

The following topics describe recommendations for the different availability requirements of your LOADPLUS jobs.

**Specifying SHRLEVEL for availability**

Specify the appropriate SHRLEVEL option for your availability requirements.

To determine the appropriate option to use and for considerations when using the SHRLEVEL option, see “SHRLEVEL considerations in LOADPLUS” on page 87.

**Granting data set authorization for availability**

For load jobs that use staging data sets, ensure that UPDATE and Control privileges are established for the data sets that LOADPLUS uses as staging data sets if both of the following conditions exist:

- You are running in a non-RACF environment
- You establish authorization at a node lower than the highest node.

**Availability when adding a relatively small amount of data**

If you are running LOAD RESUME YES for small, incremental-type loads, BMC recommends that you specify SHRLEVEL CHANGE SQLAPPLY.
In addition to providing the highest data availability, this option eliminates the need to run the following utilities after the load job:

- COPY
- CHECK, if referential integrity is involved

LOADPLUS processing phases

The topics that follow describe the processing phases of LOADPLUS in detail. This information can be useful when tuning your load jobs.

**Note**

For a DSNUTILB load, LOADPLUS passes processing to DSNUTILB after the UTILINIT phase and regains control during the UTILTERM phase. Therefore, the information in this section does not apply to a DSNUTILB load.

LOADPLUS exploits the technology that large scale processors provide. During execution, LOADPLUS examines available resources and uses as much of these resources as possible to maximize performance.

LOADPLUS architecture

The architecture of LOADPLUS differs from that of the IBM DB2 LOAD utility.

The DB2 LOAD utility performs load processing in several phases (RELOAD, SORT, BUILD, SORTBLD, INDEXVAL, ENFORCE, DISCARD, and REPORT) that run serially. LOADPLUS combines these phases into one of two architecture types.

**Two-phase architecture**

When you specify PRELOAD CONTINUE, PRELOAD PAUSE, or PRELOAD ANALYZE, LOADPLUS uses two phases, PRELOAD and LOAD. The PRELOAD and LOAD phases perform all of the functions that are included in the multiple phases performed during execution of the DB2 LOAD utility.

With its two-phase architecture, LOADPLUS performs many processes concurrently, thus reducing the elapsed time for load processing over the DB2 LOAD utility. For example, LOADPLUS checks for duplicate keys (IBM’s INDEXVAL phase) while it reads input data (IBM’s RELOAD phase) and sorts index keys (IBM’s SORT phase).
Single-phase architecture

When you specify PRELOAD LOAD or run an SQLAPPLY load, LOADPLUS combines the PRELOAD and LOAD phases into a single phase called the COMBINED phase. The COMBINED phase performs all of the functions that are included in the multiple phases performed during execution of the DB2 LOAD utility.

The LOADPLUS single-phase architecture builds on the advantages of the two-phase processing and allows for even greater reductions in CPU usage and elapsed time due to the reduction of read and write operations. For most load jobs, single-phase processing eliminates the need for work data sets (SORTOUT and SYSUT1), because LOADPLUS writes the data directly to the table space and index.

ANALYZE phase in LOADPLUS

The ANALYZE phase provides information about the number of rows (cardinality) and the average row size of the data to be loaded.

Two considerations arise from the ANALYZE phase and are controlled by the command options that you specify:

- The amount of time that the phase requires to run
- The accuracy of the information that LOADPLUS gathers during the phase

The following figure shows the objects that the ANALYZE phase uses to determine the information that it provides:

Figure 26: ANALYZE phase
PRELOAD phase

The PRELOAD phase of LOADPLUS prepares the input for loading into the specified tables by the LOAD phase.

Depending on the data that you are loading, the characteristics of the table, and the options that you specify, the PRELOAD phase performs the following tasks:

- Reads the input data from the SYSREC data set
- Generates identity column or row change timestamp column values if needed
- Converts data if needed
- Verifies that the data is correct
- Sorts the data if needed
- Sets up data to allow concurrent processing for loading table spaces and building indexes, and for concurrent loading of partitions
- Optionally sorts input data (when you specify ORDER YES)
- Unloads existing indexes and merges them with the new index keys from the input data (when you specify RESUME YES and INDEX BUILD)
  LOADPLUS sorts merged index data to build organized indexes during the LOAD phase.
- Produces a comprehensive discard report
- If you specify UNIQUECHECK YES or UNIQUECHECK CLUSTER, checks unique data-sorting keys for duplicates and discards them
- If you specify UNIQUECHECK YES, checks unique non-data-sorting indexes for duplicates and discards them
- Builds or keeps the compression dictionary and compresses the data rows
- Builds the DB2 data rows and writes them to the SORTOUT data set
- Writes the index work records to the SYSUT1 data set
- At the end of the phase, frees the allocation for the SYSREC and SYSDISC data sets
Resource allocation

The following figure shows resources that are allocated during the PRELOAD phase of a two-phase load:

Figure 27: Resource allocation during the PRELOAD phase

Read, sort/data, and index tasks

During the PRELOAD phase, LOADPLUS performs the main tasks that the following table describes:

Table 118: PRELOAD phase tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read</td>
<td>■ Reads input data</td>
</tr>
<tr>
<td></td>
<td>■ Performs data conversion</td>
</tr>
<tr>
<td></td>
<td>■ If applicable, builds the data-sorting index key</td>
</tr>
<tr>
<td>Sort/data</td>
<td>■ Sorts data rows if you specify ORDER YES</td>
</tr>
<tr>
<td></td>
<td>■ Creates non-data-sorting indexes, if any</td>
</tr>
<tr>
<td></td>
<td>■ Writes data to the SORTOUT data set</td>
</tr>
</tbody>
</table>
LOADPLUS does not perform the index task in the following cases:

- No unique indexes are participating in the load.
- You specify UNIQUECHECK NO.
- You specify UNIQUECHECK CLUSTER with ORDER YES.
- You specify INDEX UPDATE.
- Only the data-sorting index is unique and you specify ORDER YES or ORDER PRESORTED.

**Invoking BMCSORT**

LOADPLUS invokes BMCSORT during the PRELOAD phase under any of the following conditions:

- You specify the ORDER YES option on the LOAD command.
- You specify UNIQUECHECK CLUSTER and unique data-sorting indexes exist.
- You specify UNIQUECHECK YES and unique indexes are participating in the load.
- The table space is segmented and contains multiple tables.
- LOADPLUS discards any records for duplicate key violations.

**LOAD phase**

The LOAD phase of LOADPLUS is basically an I/O driver set up by the PRELOAD phase.

During the LOAD phase, LOADPLUS performs the following basic functions:

- Loads data into the specified tables and creates image copies if requested
- Builds all associated participating indexes

Depending on the data that you are loading, the characteristics of the objects, and the options that you specify, LOADPLUS performs the following tasks during the LOAD phase:
- Adds partitions for partition-by-growth table spaces if required (for SHRLEVEL NONE)

- Redefines the VSAM data sets (including staging data sets) when the value of the REDEFINE command or installation option is YES

- Concurrently loads multiple partitions

- Concurrently loads the table space and data-sorting indexes

- Concurrently loads the table space and sorts and builds non-data-sorting indexes

- Collects statistics while loading table spaces and building indexes

- Produces image copies either as the data is loaded or after the data is loaded

- Detects duplicates and writes error records out to SYSERR

- After loading table spaces and building indexes, resolves any duplicates that are detected during the LOAD phase

**Resource allocation**

The following figure shows resources that are allocated during the LOAD phase of a two-phase load:

*Figure 28: Resource allocation during the LOAD phase*
**Load process and index-building process**

The LOAD phase is made up of one or more load processes and one or more index-building processes. LOADPLUS performs these processes concurrently. The execution strategy of the LOAD phase depends on the characteristics of the table space that you are loading and the options used during the PRELOAD phase.

**Invoking BMCSORT**

LOADPLUS invokes BMCSORT during the LOAD phase under any of the following conditions:

- You specify UNIQUECHECK CLUSTER and unique non-data-sorting indexes are participating in the load.
- You specify UNIQUECHECK NO and unique indexes are participating in the load.
- The job builds non-unique non-data-sorting indexes.
- The job has discards.

**COMBINED phase for a single-phase load**

During the COMBINED phase for a single-phase load, LOADPLUS performs almost all of the same processes as the PRELOAD and LOAD phases.

*Note*

This book distinguishes between the COMBINED phase for a single-phase load (in which you specify PRELOAD LOAD) and the COMBINED phase for an SQLAPPLY load. For more information about the latter type of COMBINED phase, see “COMBINED phase for SQLAPPLY” on page 698.

Depending on the data that you are loading, the characteristics of the objects, and the options that you specify, LOADPLUS performs the following tasks during the COMBINED phase:

- Reads the input data from the SYSREC data set
- Generates identity column or row change timestamp column values if needed
- Verifies that the data is correct
- For certain types of single-phase jobs, writes index key entries to SYSUT1 and data-sorting key entries and the new DB2 row images to SORTOUT, if the data sets are allocated
- Redefines the VSAM data sets (including staging data sets) when the value of the REDEFINE command or installation option is YES

- Sets up data to allow concurrent processing for loading table spaces and building indexes, and for concurrent loading of partitions

- Optionally sorts input data (when you specify ORDER YES)

- Unloads existing indexes and merges them with the new index keys from the input data (when you specify RESUME YES and INDEX BUILD)

- Sorts merged indexes and writes them directly to the index data set

- Produces a comprehensive discard report

- Checks unique data-sorting keys for duplicates and deletes them from the table and index space

- Checks participating unique non-data-sorting indexes for duplicates and deletes the duplicates from the table and index space

- Verifies referential constraints and deletes rows that are in violation of the constraints

- Builds or keeps the compression dictionary and compresses the data rows

- Builds the DB2 data rows and writes them directly to the table space

- Concurrently loads multiple partitions

- Concurrently loads the table space and data-sorting indexes

- Collects statistics while loading table spaces and building indexes

- Produces image copies either as the data loads or after the data is loaded

- At the end of the phase, frees the allocation for the SYSREC and SYSDISC data sets
Resource allocation

The following figure shows resources that are allocated during the COMBINED phase of a single-phase load:

Figure 29: Resource allocation during the COMBINED phase for single-phase load

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read</td>
<td>■ Reads input data</td>
</tr>
<tr>
<td></td>
<td>■ Performs data conversion</td>
</tr>
<tr>
<td></td>
<td>■ If applicable, builds the data-sorting index key</td>
</tr>
</tbody>
</table>

Read, sort/data, and index tasks

The following table describes the main tasks that LOADPLUS performs during the COMBINED phase of a single-phase load:

Table 119: COMBINED phase tasks for single-phase load
<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sort/data</td>
<td>■ Sorts data rows when you specify ORDER YES</td>
</tr>
<tr>
<td></td>
<td>■ If applicable, writes data-sorting index records to the index space when you specify ORDER YES</td>
</tr>
<tr>
<td></td>
<td>■ Builds non-data-sorting index keys, if any, and passes them to the index task</td>
</tr>
<tr>
<td></td>
<td>■ Writes data to the table spaces</td>
</tr>
<tr>
<td>Index</td>
<td>■ Writes index records to the index space</td>
</tr>
</tbody>
</table>

LOADPLUS does not perform the index task in the following cases:

■ No indexes are participating in the load.
■ There is a single index participating in the load and it is built with the data task.

**Invoking BMCSORT**

LOADPLUS always invokes BMCSORT during the COMBINED phase of a single-phase load *except* when both of the following conditions occur:

■ No records are discarded.
■ No indexes are participating in the load, or you specify ORDER PRESORTED and the only participating indexes are built with the data.

**COMBINED phase for SQLAPPLY**

During the COMBINED phase for an SQLAPPLY load, LOADPLUS performs most of the same processes as in the PRELOAD and LOAD phases.

Depending on the data that you are loading, the characteristics of the objects, and the options that you specify, LOADPLUS performs the following tasks during the COMBINED phase:

■ Reads the input data from the SYSREC data set
■ Verifies that the data is correct, but does not perform checking for duplicates or check constraints
**Note**

DB2 handles checking for duplicates and check constraints during apply processing.

- Sets up data to allow concurrent processing for loading tables and partitions
- Optionally sorts input data (when you specify ORDER YES)
- Passes the data to High-speed Apply, for inserting into the table space and indexes
- Produces a comprehensive discard report
- At the end of the phase, frees the allocation for the SYSREC and SYSDISC data sets
Resource allocation

The following figure shows resources that are allocated during the COMBINED phase for an SQLAPPLY load:

**Figure 30: Resource allocation during the COMBINED phase for SQLAPPLY**

Read, sort, and SQLAPPLY tasks

The following table shows the main tasks that LOADPLUS performs during the COMBINED phase of an SQLAPPLY load:
Table 120: COMBINED phase tasks for SQLAPPLY

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read</td>
<td>■ Reads input data</td>
</tr>
<tr>
<td></td>
<td>■ Performs data conversion</td>
</tr>
<tr>
<td>Sort</td>
<td>Sorts data rows if you specify ORDER YES</td>
</tr>
<tr>
<td>SQLAPPLY</td>
<td>Passes data to High-speed Apply for loading</td>
</tr>
</tbody>
</table>

Invoking BMCSORT

LOADPLUS invokes BMCSORT during the COMBINED phase of an SQLAPPLY load if you specify ORDER YES and you have discards.
LOADPLUS installation options

This chapter describes the options that you can specify in your installation options modules.

Overview of LOADPLUS installation options

Options set during installation establish default values that LOADPLUS uses during your load job.

The LOADPLUS for DB2 product is installed by using the BMC Installation System. During this installation, the configuration process generates a customized installation data set. This data set contains customized jobs that install LOADPLUS into your specific DB2 environment. Two of these jobs establish the default processing option values that LOADPLUS uses:

- $730DOP2 establishes the defaults for LOADPLUS and High-speed Apply processing options.
- $532SOPT contains options for the BMC BMCSORT technology.

These jobs assemble options macros. The macros contain the LOADPLUS processing options and the values for those options that are shipped with LOADPLUS and BMCSORT. When the Installation System-generated configuration job is submitted, it links the AMU$OPTS installation options module in the APF-authorized library that is designated by your site.

You can customize the installation of LOADPLUS by changing the values for the LOADPLUS installation options. However, if you change any of the values in $730DOP2 or $532SOPT after LOADPLUS 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 LOADPLUS. 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 “Installation options...”
module” on page 431. For more information about customizing your installation of LOADPLUS, see the BMC Products and Solutions for DB2 Customization Guide.

To find a description of each options macro, use the following table:

Table 121: LOADPLUS installation macros

<table>
<thead>
<tr>
<th>Job</th>
<th>Macro name</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>$730DOP2</td>
<td>$AMUOPTS</td>
<td>Basic options</td>
<td>“Basic LOADPLUS installation options” on page 704</td>
</tr>
<tr>
<td></td>
<td>$AMUDYNA</td>
<td>Options for dynamic data set allocation</td>
<td>“Dynamic allocation installation options for LOADPLUS” on page 760</td>
</tr>
<tr>
<td>$532SOPT</td>
<td>$AUPSMAC</td>
<td>Options for BMCSORT</td>
<td>“DYNALOC installation option” on page 783</td>
</tr>
</tbody>
</table>

**Basic LOADPLUS installation options**

This topic lists the options contained in the $AMUOPTS macro in $730DOP2.

For each option, the table in this topic provides the value that ships with this version of LOADPLUS (or No value if the option is shipped without a value), a brief description, and a reference to more details. If an option ships with no value, the table shows a recommended value or example value.

Table 122: Basic LOADPLUS installation options

<table>
<thead>
<tr>
<th>Option</th>
<th>Shipped value</th>
<th>Brief description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>APCOLLECTION</td>
<td>No value</td>
<td>Default collection ID for SQLAPPLY load jobs</td>
<td>“APCOLLECTION=” on page 712</td>
</tr>
<tr>
<td></td>
<td>Example value: APTAB6S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>APCOMMIT</td>
<td>2500</td>
<td>Number of records between commits for SQLAPPLY load jobs</td>
<td>“APCOMMIT=2500” on page 712</td>
</tr>
<tr>
<td>APDOPTS</td>
<td>No value</td>
<td>For future use</td>
<td>“APDOPTS=” on page 713</td>
</tr>
<tr>
<td>APMULTIROW</td>
<td>100</td>
<td>Number of rows inserted during a single insert action for SQLAPPLY load jobs</td>
<td>“APMULTIROW=100” on page 713</td>
</tr>
<tr>
<td>Option</td>
<td>Shipped value</td>
<td>Brief description</td>
<td>Reference</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------</td>
<td>------------------------------------------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>APMXAGNT</td>
<td>10</td>
<td>Maximum number of agents to start for SQLAPPLY load jobs</td>
<td>“APMXAGNT=10” on page 713</td>
</tr>
<tr>
<td>APOWERER</td>
<td>No value</td>
<td>Default authorization ID to use to bind the DB2 plan and packages for SQLAPPLY load jobs</td>
<td>“APowerer=” on page 714</td>
</tr>
<tr>
<td>APRETLIM</td>
<td>COUNT</td>
<td>Type of retry to use after a SQL error during apply processing (in conjunction with APRETVAL)</td>
<td>“APRETLIM=COUNT” on page 714</td>
</tr>
<tr>
<td>APRETVAL</td>
<td>5</td>
<td>Value of the type of retry to use after a SQL error during apply processing (in conjunction with APRETLIM)</td>
<td>“APRETVAl=5” on page 715</td>
</tr>
<tr>
<td>AUTOENUMROWS</td>
<td>YES</td>
<td>Whether to enable automated file size estimation</td>
<td>“AUTOENUMROWS=YES” on page 715</td>
</tr>
<tr>
<td>AVAILPAGEPCT</td>
<td>0</td>
<td>Percentage of available 4-KB pages that the system reports and that LOADPLUS can allocate to BMCSORT for sort processing</td>
<td>“AVAILPAGEPCT=0” on page 716</td>
</tr>
<tr>
<td>CBUFFS</td>
<td>30</td>
<td>Multiplier that controls the amount of buffer pool storage for each copy data set</td>
<td>“CBUFFS=30” on page 717</td>
</tr>
<tr>
<td>CENTURY</td>
<td>(1950,2049)</td>
<td>Century specification for two-digit years</td>
<td>“CENTURY=(1950,2049)” on page 717</td>
</tr>
<tr>
<td>CHEKPEND</td>
<td>(YES,)</td>
<td>Whether to set CHECK pending when needed</td>
<td>“CHEKPEND=(YES,)” on page 717</td>
</tr>
<tr>
<td>COPYDDN</td>
<td>(BMCCPY,BMCCPZ)</td>
<td>Default ddname or prefix for local primary and secondary copy data sets</td>
<td>“COPYDDN=(BMCCPY, BMCCPZ)” on page 718</td>
</tr>
<tr>
<td>COPYLVL</td>
<td>FULL</td>
<td>How to dynamically allocate copy data sets for partitioned table spaces</td>
<td>“COPYLVL=FULL” on page 719</td>
</tr>
<tr>
<td>COPYPEND</td>
<td>(YES,)</td>
<td>Whether to set COPY pending when needed</td>
<td>“COPYPEND=(YES,)” on page 719</td>
</tr>
<tr>
<td>Option</td>
<td>Shipped value</td>
<td>Brief description</td>
<td>Reference</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>COPYSUBSET</td>
<td>NO</td>
<td>Whether to allow a single copy of a subset of partitions</td>
<td>“COPYSUBSET=NO” on page 720</td>
</tr>
<tr>
<td>DELFILES</td>
<td>(YES,NO)</td>
<td>Whether to delete work files on completion</td>
<td>“DELFILES=(YES,NO)” on page 720</td>
</tr>
<tr>
<td>DISCARDLIMRC</td>
<td>8</td>
<td>Return code to assign when the discard limit is reached</td>
<td>“DISCARDLIMRC=8” on page 722</td>
</tr>
<tr>
<td>DISCARDRC</td>
<td>0</td>
<td>Return code to assign on successful job completion when there are discards</td>
<td>“DISCARDRC=0” on page 722</td>
</tr>
<tr>
<td>DISCDDN</td>
<td>SYSDISC</td>
<td>Default ddname for the discard data set</td>
<td>“DISCDDN=SYSDISC” on page 723</td>
</tr>
<tr>
<td>DRNDELAY</td>
<td>1</td>
<td>Minimum number of seconds to wait before the next attempt to obtain a drain</td>
<td>“DRNDELAY=1” on page 723</td>
</tr>
<tr>
<td>DRNRETRY</td>
<td>255</td>
<td>Maximum number of times to attempt to obtain a drain before terminating</td>
<td>“DRNRETRY=255” on page 723</td>
</tr>
<tr>
<td>DRNWAIT</td>
<td>NONE</td>
<td>Timeout value for object drains</td>
<td>“DRNWAIT=NONE” on page 724</td>
</tr>
<tr>
<td>DSNUEXIT</td>
<td>(NONE,ASM)</td>
<td>Name of the user exit for creating dynamically allocated data set names and the language it is written in</td>
<td>“DSNUEXIT=(NONE,ASM)” on page 725</td>
</tr>
<tr>
<td>DSNUTILB</td>
<td>YES</td>
<td>Whether to use DSNUTILB to process the load job when a feature requires it</td>
<td>“DSNUTILB=YES” on page 725</td>
</tr>
<tr>
<td>ENFORCE</td>
<td>CHECK</td>
<td>Whether to check DB2-defined constraints</td>
<td>“ENFORCE=CHECK” on page 726</td>
</tr>
<tr>
<td>ERRDDN</td>
<td>SYSERR</td>
<td>Default ddname for the error data set</td>
<td>“ERRDDN=SYSERR” on page 727</td>
</tr>
<tr>
<td>EXCLUDUMP</td>
<td>(X37,X22,X06)</td>
<td>System codes to ignore during the dump process</td>
<td>“EXCLUDUMP=(X37,X22,X06)” on page 727</td>
</tr>
<tr>
<td>FASTSWITCH</td>
<td>NO</td>
<td>Whether to bypass the VSAM rename process and directly update the DB2 catalog to use the staging data set names</td>
<td>“FASTSWITCH=NO” on page 727</td>
</tr>
<tr>
<td>Option</td>
<td>Shipped value</td>
<td>Brief description</td>
<td>Reference</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------</td>
<td>------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>FILECHK</td>
<td>FAIL</td>
<td>Which action to take when encountering a temporary work file</td>
<td>“FILECHK=FAIL” on page 728</td>
</tr>
<tr>
<td>FILEVAR</td>
<td>Not applicable</td>
<td>Obsolete</td>
<td>“FILEVAR (obsolete)” on page 729</td>
</tr>
<tr>
<td>FORCE</td>
<td>NONE</td>
<td>Whether to cancel DB2 threads that might prevent a drain process from completing</td>
<td>“FORCE=None” on page 729</td>
</tr>
<tr>
<td>FORCE_AT</td>
<td>(START,3)</td>
<td>When to cancel DB2 threads that might prevent the drain process from completing</td>
<td>“FORCE_AT=(START,3)” on page 729</td>
</tr>
<tr>
<td>FORCE_RPT</td>
<td>NO</td>
<td>Whether to display a report of the canceled threads</td>
<td>“FORCE_RPT=NO” on page 730</td>
</tr>
<tr>
<td>HASHAX</td>
<td>Not applicable</td>
<td>Obsolete</td>
<td>“HASHAX (obsolete)” on page 731</td>
</tr>
<tr>
<td>IBUFFS</td>
<td>20</td>
<td>Multiplier that controls the amount of buffer pool storage for each input data set</td>
<td>“IBUFFS=20” on page 731</td>
</tr>
<tr>
<td>IDCACHE</td>
<td>1000</td>
<td>Cache size of identity column numbers</td>
<td>“IDCACHE=1000” on page 732</td>
</tr>
<tr>
<td>IDCDDN</td>
<td>SYSIDCIN</td>
<td>Default ddname for the data set that contains IDCAMS commands</td>
<td>“IDCDDN=SYSIDCIN” on page 732</td>
</tr>
<tr>
<td>IDERROR</td>
<td>DISCARD</td>
<td>Which action to take when encountering an error on the identity column range</td>
<td>“IDERROR=Discard” on page 732</td>
</tr>
<tr>
<td>INDDN</td>
<td>SYSREC</td>
<td>Default ddname or prefix for input data sets</td>
<td>“INDDN=SYSREC” on page 733</td>
</tr>
<tr>
<td>INFORI</td>
<td>NO</td>
<td>Whether to enforce informational referential constraints</td>
<td>“INFORI=NO” on page 733</td>
</tr>
<tr>
<td>INLINECP</td>
<td>YES</td>
<td>Whether to create an inline image copy</td>
<td>“INLINECP=YES” on page 734</td>
</tr>
<tr>
<td>INLOB</td>
<td>Not applicable</td>
<td>Obsolete</td>
<td>“INLOB (obsolete)” on page 734</td>
</tr>
<tr>
<td>IXONEX</td>
<td>Not applicable</td>
<td>Obsolete</td>
<td>“IXONEX (obsolete)” on page 735</td>
</tr>
<tr>
<td>Option</td>
<td>Shipped value</td>
<td>Brief description</td>
<td>Reference</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------</td>
<td>-------------------------------------------------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>IXRANDOM</td>
<td>Not applicable</td>
<td>Obsolete</td>
<td>“IXRANDOM (obsolete)” on page 735</td>
</tr>
<tr>
<td>KEEPDICTATION</td>
<td>NO</td>
<td>Whether to keep an existing compression dictionary or build a new one</td>
<td>“KEEPDICTATION=NO” on page 735</td>
</tr>
<tr>
<td>LBUFFS</td>
<td>20</td>
<td>Multiplier that controls the amount of buffer pool storage for each load data set</td>
<td>“LBUFFS=20” on page 736</td>
</tr>
<tr>
<td>LOADDN</td>
<td>SORTOUT</td>
<td>Default ddname or prefix for load data sets</td>
<td>“LOADDN=SORTOUT” on page 736</td>
</tr>
<tr>
<td>LOB</td>
<td>Not applicable</td>
<td>Obsolete</td>
<td>“LOB (obsolete)” on page 737</td>
</tr>
<tr>
<td>LOBAVGPCCT</td>
<td>50</td>
<td>Estimate of the percentage of the maximum size defined for the LOB</td>
<td>“LOBAVGPCCT=50” on page 737</td>
</tr>
<tr>
<td>LOCKROW</td>
<td>YES</td>
<td>Serialization method for BMCSYNC and BMCUTIL</td>
<td>“LOCKROW=YES” on page 737</td>
</tr>
<tr>
<td>LONGNAMETRUNC</td>
<td>MIDDLE (or M)</td>
<td>Long name truncation method for messages</td>
<td>“LONGNAMETRUNC=MIDDLE” on page 738</td>
</tr>
<tr>
<td>MAPDDN</td>
<td>SYSMAP</td>
<td>Default ddname for the mapping data set for a DSNUTILB load</td>
<td>“MAPDDN=SYSMAP” on page 738</td>
</tr>
<tr>
<td>MAXP</td>
<td>5</td>
<td>Maximum number of partitions to load concurrently</td>
<td>“MAXP=5” on page 739</td>
</tr>
<tr>
<td>MAXSORTMEMORY</td>
<td>0</td>
<td>Maximum amount of memory to allocate to each sort task</td>
<td>“MAXSORTMEMORY=0” on page 739</td>
</tr>
<tr>
<td>MAXTAPE</td>
<td>3</td>
<td>Maximum number of tape units to allocate dynamically</td>
<td>“MAXTAPE=3” on page 739</td>
</tr>
<tr>
<td>MGEXTENT</td>
<td>CONTINUE</td>
<td>How to allocate extents when extending to a new data set</td>
<td>“MGEXTENT=CONTINUE” on page 740</td>
</tr>
<tr>
<td>MINSORTMEMORY</td>
<td>0</td>
<td>Minimum amount of memory to allocate to each sort task</td>
<td>“MINSORTMEMORY=0” on page 740</td>
</tr>
<tr>
<td>Option</td>
<td>Shipped value</td>
<td>Brief description</td>
<td>Reference</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>MSGLEVEL</td>
<td>1</td>
<td>Default for the message-level execution parameter</td>
<td>“MSGLEVEL=1” on page 741</td>
</tr>
<tr>
<td>OPNDB2ID</td>
<td>YES</td>
<td>Whether to use the user’s RACF ID instead of the DB2 RACF ID</td>
<td>“OPNDB2ID=YES” on page 741</td>
</tr>
<tr>
<td>ORIGDISP</td>
<td>DELETE</td>
<td>Disposition of the original data set during the staging data set rename or FASTSWITCH process</td>
<td>“ORIGDISP=DELETE” on page 741</td>
</tr>
<tr>
<td>PAUSEDISCARDRC</td>
<td>4</td>
<td>Return code to assign when pausing after the PRELOAD phase when there are discards</td>
<td>“PAUSEDISCARDRC=4” on page 742</td>
</tr>
<tr>
<td>PLAN</td>
<td>AMU1120</td>
<td>Product plan name</td>
<td>“PLAN=AMU1120” on page 742</td>
</tr>
<tr>
<td>PREFORMAT</td>
<td>NO</td>
<td>Whether to preformat unused data set pages</td>
<td>“PREFORMAT=NO” on page 743</td>
</tr>
<tr>
<td>PRELOAD</td>
<td>LOAD</td>
<td>Whether to perform single-phase or two-phase processing</td>
<td>“PRELOAD=LOAD” on page 743</td>
</tr>
<tr>
<td>RCVYDDN</td>
<td>(BMCRCY,BMCRCZ)</td>
<td>Default ddname or prefix for remote primary and secondary copy data sets</td>
<td>“RCVYDDN=(BMCRCY, BMCRCZ)” on page 744</td>
</tr>
<tr>
<td>REDEFINE</td>
<td>YES</td>
<td>Whether to delete and redefine the VSAM data sets</td>
<td>“REDEFINE=YES” on page 744</td>
</tr>
<tr>
<td>RENMMAX</td>
<td>30</td>
<td>Maximum number of tasks to start during the staging data set rename process</td>
<td>“RENMMAX=30” on page 745</td>
</tr>
<tr>
<td>RULES</td>
<td>STANDARD</td>
<td>Which set of comparison rules to use in WHEN, NULLIF, and DEFAULTIF processing</td>
<td>“RULES=STANDARD” on page 745</td>
</tr>
<tr>
<td>SDUMP</td>
<td>(ALLPSA,CSA,RGN,SQA,LSQA,SUM,TRT,IO)</td>
<td>System dump parameters</td>
<td>“SDUMP=(ALLPSA,CSA,RGN,SQA,LSQA,SUM,TRT,IO)” on page 746</td>
</tr>
<tr>
<td>SHORTMEMORY</td>
<td>CONTINUE</td>
<td>Which action to take when a memory shortage exists</td>
<td>“SHORTMEMORY=CONTINUE” on page 746</td>
</tr>
<tr>
<td>SHRLEVEL</td>
<td>(NONE,NONE)</td>
<td>SHRLEVEL option default</td>
<td>“SHRLEVEL=(NONE,NONE)” on page 747</td>
</tr>
<tr>
<td>Option</td>
<td>Shipped value</td>
<td>Brief description</td>
<td>Reference</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------</td>
<td>------------------------------------------------------------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>SMAX</td>
<td>16</td>
<td>Maximum number of each type of task to invoke concurrently</td>
<td>“SMAX=16” on page 749</td>
</tr>
<tr>
<td>SMCORE</td>
<td>(0K,0K)</td>
<td>Maximum amount of sort memory</td>
<td>“SMCORE=(0K,0K)” on page 749</td>
</tr>
<tr>
<td>SORTDEVT</td>
<td>(SYSALLDA)</td>
<td>Sort device types</td>
<td>“SORTDEVT=(SYSALLDA)” on page 750</td>
</tr>
<tr>
<td>SORTNUM</td>
<td>32</td>
<td>Number of sort work files per sort task for BMCSORT to allocate dynamically</td>
<td>“SORTNUM=32” on page 751</td>
</tr>
<tr>
<td>SQLDELAY</td>
<td>3</td>
<td>Number of seconds between retry attempts after SQLOCODE -911</td>
<td>“SQLDELAY=3” on page 752</td>
</tr>
<tr>
<td>SQLRETRY</td>
<td>100</td>
<td>Number of SQL attempts</td>
<td>“SQLRETRY=100” on page 752</td>
</tr>
<tr>
<td>STOP@CMT</td>
<td>YES</td>
<td>Whether to add the AT(COMMIT) parameter to all DB2 STOP commands that LOADPLUS issues</td>
<td>“STOP@CMT=YES” on page 752</td>
</tr>
<tr>
<td>STOPDELAY</td>
<td>1</td>
<td>Number of seconds between checking to determine whether DB2 has stopped the object</td>
<td>“STOPDELAY=1” on page 752</td>
</tr>
<tr>
<td>STOPRETRY</td>
<td>300</td>
<td>Number of times to check to determine whether DB2 has stopped the object</td>
<td>“STOPRETRY=300” on page 753</td>
</tr>
<tr>
<td>TAPEDISP</td>
<td>DELETE</td>
<td>Tape file disposition</td>
<td>“TAPEDISP=DELETE” on page 753</td>
</tr>
<tr>
<td>TEMPRALDATA</td>
<td>Not applicable</td>
<td>Obsolete</td>
<td>“TEMPRALDATA (obsolete)” on page 753</td>
</tr>
<tr>
<td>TOTALPAGEPCT</td>
<td>30</td>
<td>Percentage of total 4-KB pages that the system reports and that LOADPLUS can allocate to BMCSORT for sort processing</td>
<td>“TOTALPAGEPCT=30” on page 753</td>
</tr>
<tr>
<td>TSSAMPLEPCT</td>
<td>100</td>
<td>Percentage of sampling to perform during statistics gathering</td>
<td>“TSSAMPLEPCT=100” on page 754</td>
</tr>
<tr>
<td>Option</td>
<td>Shipped value</td>
<td>Brief description</td>
<td>Reference</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>UNICODE</td>
<td>Not applicable</td>
<td>Obsolete</td>
<td>“UNICODE (obsolete)” on page 755</td>
</tr>
<tr>
<td>UPDMAXA</td>
<td>NO</td>
<td>Whether to update MAXASSIGNEDVAL when loading identity column values</td>
<td>“UPDMAXA=NO” on page 755</td>
</tr>
<tr>
<td>UPDMAXA_AUTHID</td>
<td>USER</td>
<td>Which authorization ID to use to provide ALTER privileges to update MAXASSIGNEDVAL</td>
<td>“UPDMAXA_AUTHID=USER” on page 756</td>
</tr>
<tr>
<td>UTILB_COLCCSID</td>
<td>Not applicable</td>
<td>Obsolete</td>
<td>“UTILB_COLCCSID (obsolete)” on page 756</td>
</tr>
<tr>
<td>UTILB_NULLIX</td>
<td>Not applicable</td>
<td>Obsolete</td>
<td>“UTILB_NULLIX (obsolete)” on page 756</td>
</tr>
<tr>
<td>UXSTATE</td>
<td>SUP</td>
<td>How to invoke DB2 user exits</td>
<td>“UXSTATE=SUP” on page 756</td>
</tr>
<tr>
<td>WBUFFS</td>
<td>(20,10)</td>
<td>Multiplier that controls the amount of buffer pool storage for each work data set</td>
<td>“WBUFFS=(20,10)” on page 757</td>
</tr>
<tr>
<td>WORKDDN</td>
<td>SYSUT1</td>
<td>Default ddname or prefix for work data sets</td>
<td>“WORKDDN=SYSUT1” on page 757</td>
</tr>
<tr>
<td>WORKUNIT</td>
<td>SYSALLDA</td>
<td>Temporary unit for work data sets</td>
<td>“WORKUNIT=SYSALLDA” on page 757</td>
</tr>
<tr>
<td>XBLKS</td>
<td>3</td>
<td>Number of blocks to use for transferring data between tasks</td>
<td>“XBLKS=3” on page 757</td>
</tr>
<tr>
<td>XBMID</td>
<td>No value</td>
<td>XBM subsystem that LOADPLUS accesses when it uses XBM or SUF</td>
<td>“XBMID=“ on page 758</td>
</tr>
<tr>
<td>XML</td>
<td>Not applicable</td>
<td>Obsolete</td>
<td>“XML (obsolete)” on page 758</td>
</tr>
<tr>
<td>XMLAVGSIZE</td>
<td>10M</td>
<td>Average size of the XML input data</td>
<td>“XMLAVGSIZE=10M” on page 759</td>
</tr>
<tr>
<td>ZEROROWRC</td>
<td>0</td>
<td>Return code to assign when no rows are loaded</td>
<td>“ZEROROWRC=0” on page 759</td>
</tr>
<tr>
<td>ZIIP</td>
<td>ENABLED</td>
<td>Whether to enable zIIP processing</td>
<td>“ZIIP=ENABLED” on page 759</td>
</tr>
</tbody>
</table>
Descriptions of basic LOADPLUS installation options

This topic describes each of the basic LOADPLUS installation options.

For more information about setting the values of some of these options at runtime, see “LOADPLUS command options” on page 182. For more information about the performance implications of these options, see “Tuning LOADPLUS jobs” on page 645.

APCOLLECTION=

This option is valid only for SQLAPPLY load jobs.

The APCOLLECTION option specifies the ID for the collection to which High-speed Apply dynamically binds packages during execution. If you change the value for this installation option, you can specify a collection ID with a length of up to 128 bytes. LOADPLUS passes the value of this option to High-speed Apply.

Overriding this option

You can override the value for this option by using the APCOLLECTION command option.

If you keep the null value that was shipped with LOADPLUS and no value is specified for the APCOLLECTION command option, High-speed Apply uses the default value of the CollectionID parameter in the Bind section of High-speed Apply. You cannot specify a null value for the APCOLLECTION command option.

APCOMMIT=2500

This option is valid only for SQLAPPLY load jobs.

The APCOMMIT option specifies the number of records that you want each High-speed Apply agent to load before issuing a COMMIT statement. Valid values for this option are 1 through 32767.

LOADPLUS passes the value of this parameter to the StatementCount parameter in the CommitTriggers section of High-speed Apply. For more information, see the High-speed Apply Engine Reference Manual.

Overriding this option

You can override the value for this option by using the APCOMMIT command option.
**APDOPTS=**

This option will be used in a future release.

**APMULTIROW=100**

*This option is valid only for SQLAPPLY load jobs.*

The APMULTIROW option tells High-speed Apply whether to insert multiple rows in a single insert action and, if so, how many rows to insert at one time. You can specify a value from 0 through 32767.

A value of 0 or 1 tells High-speed Apply to insert one row at a time.

LOADPLUS passes this value to the MultiRowInsert and MaxRows parameters in the MultiRowInsert section of High-speed Apply. For more information about these parameters, see the *High-speed Apply Engine Reference Manual.*

**Overriding this option**

You can override the value for this option by using the APMULTIROW command option.

**APMXAGNT=10**

*This option is valid only for SQLAPPLY load jobs.*

The APMXAGNT option specifies the maximum number of agents that High-speed Apply can start for this load. Valid values for this option are 1 through the number of batch threads that are available in your DB2 subsystem.

APMXAGNT affects only partitioned or multi-table table spaces. LOADPLUS uses at most one agent per partition and table. For partition-by-growth table spaces, LOADPLUS uses only one agent per load regardless of the value that you specify.

LOADPLUS passes the value of this parameter to the MaxAgents parameter in the Agent section of High-speed Apply. For more information, see the *High-speed Apply Engine Reference Manual.*
Note
If you specify a value greater than 1 when the following conditions exist, you might encounter a task deadlock, which could result in the LOADPLUS job failing:

- You are loading a partitioned table space (except partition-by-growth).
- The table space is defined with LOCKSIZE TABLESPACE or LOCKSIZE ANY.

If you encounter this deadlock condition, BMC recommends that you change the value to 1.

Overriding this option
You can override the value for this option by using the APMAXAGENTS command option.

APOWNER=

This option is valid only for SQLAPPLY load jobs.

The APOWNER option specifies the authorization ID that High-speed Apply uses to bind the DB2 plan and packages for the apply request. If you change the value for this installation option, you can specify an owner ID with a length of up to 128 bytes. LOADPLUS passes this value to High-speed Apply.

Overriding this option
You can override the value for this option by using the APOWNER command option.

If you keep the null value that was shipped with LOADPLUS and do not specify a value for the APOWNER command option, High-speed Apply uses the default value of the BindOwner parameter in the Bind section of High-speed Apply. You cannot specify a null value for the APOWNER command option.

APRETLIM=COUNT

This option is valid only for SQLAPPLY load jobs.

The APRETLIM option tells High-speed Apply how to respond to an SQL -911, -913, or -904 error. You can specify one of the following options:

- COUNT tells High-speed Apply to retry the number of times specified in the APRETVL option.
- TIME tells High-speed Apply to continue to retry until it reaches the number of seconds specified in APRETVL.
LOADPLUS passes the value of this parameter to the RetryLimit parameter in the Conflict section of High-speed Apply. For more information, see the *High-speed Apply Engine Reference Manual*.

**Overriding this option**

You can override the value for this option by using the APRETRYLIM command option.

**APRETRYVAL=5**

*This option is valid only for SQLAPPLY load jobs.*

The APRETRYVAL option specifies the High-speed Apply retry limit following an SQL -911, -913, or -904 error. Depending on the value for APRETLIM, this limit is either the number of retry attempts or the number of seconds after which High-speed Apply is to stop retrying. Valid values are 0 through 32767. A value of 0 tells High-speed Apply not to retry.

LOADPLUS passes the value of this parameter to the RetryValue parameter in the Conflict section of High-speed Apply. For more information, see the *High-speed Apply Engine Reference Manual*.

**Overriding this option**

You can override the value for this option by using the APRETRYVAL command option.

**AUTOENUMROWS=YES**

The AUTOENUMROWS option specifies whether to automate input file size estimation for dynamic allocation and sort processing optimization. Valid values for this option are YES and NO.

YES automates file size estimation. Automating file size estimation eliminates the need to provide an estimate with the ENUMROWS command option when enabling dynamic data set allocation or specifying the ANALYZE command option. For more information about automated file size estimation, see “ENUMROWS” on page 229.

If you specify NO, the following conditions and requirements apply:

- If you do not specify a value for the ENUMROWS command option, the memory allocated for each sort process could be limited.

- To enable dynamic allocation (except for a DSNUTILB load) or if you specify the ANALYZE command option, you must specify a value for the ENUMROWS command option.
When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the ENUMROWS command option.

**AVAILPAGEPCT=0**

The AVAILPAGEPCT option controls memory that LOADPLUS allocates to BMCSORT for concurrent sort processing. AVAILPAGEPCT specifies the maximum percentage of available 4-KB pages, as obtained from the system, that LOADPLUS can allocate. You can specify any integer from 0 through 100.

**Note**

LOADPLUS defines *available pages* as pages that have not been used. *Total pages* (which you can control with the TOTALPAGEPCT option), are pages that are underutilized and are available for use.

A value of 0 tells LOADPLUS to ignore the number of available pages when allocating sort memory.

A value of 1 through 100 tells LOADPLUS to use up to the specified percentage of available pages when allocating sort memory. For example, AVAILPAGEPCT=50 indicates that LOADPLUS should use no more than 50 percent of the available pages.

**Additional considerations**

The following additional information applies to the AVAILPAGEPCT option:

- Because available pages are rarely subject to system paging, changing this value will have a minimal effect, if any, on system performance.

- When you specify values greater than 0 for both AVAILPAGEPCT and TOTALPAGEPCT, LOADPLUS uses the lesser of the two calculated results as the maximum amount of memory for sort processing.

- If LOADPLUS is unable to perform an optimal sort due to an insufficient number of available or total pages that it is enabled to allocate, the SHORTMEMORY installation or command option controls the action that LOADPLUS takes.

- When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**
You can override the value for this option by using the AVAILPAGEPCT command option.

**CBUFFS=30**

The CBUFFS option specifies a multiple of 32 KB to use to define the amount of buffer pool storage for each copy data set. LOADPLUS allocates as many buffers as will fit into the calculated space. If you are making multiple copies, the same set of buffers is used, not an additional set. For more information, see “Copy data sets in LOADPLUS” on page 436.

**CENTURY=(1950,2049)**

The CENTURY option specifies the 100-year range that determines the century for the DATE and TIMESTAMP external formats that contain two-digit year values. The value for this option is two four-digit years in the format (ccyy,ccyy). The first four-digit year value must be less than the second four-digit year. You must specify both values, and these values must span 100 years.

Any two-digit year in your input data that lies between the first yy specification and 99 is prefixed with the first cc value to create a four-digit year. Any two-digit year in your input data that lies between 00 and the second yy specification is prefixed with the second cc value to create a four-digit year.

For example, if you specify CENTURY(1950,2049), LOADPLUS places 19 in front of each two-digit year with a value 50 through 99, and places 20 in front of each two-digit year with a value 00 through 49. The date 99/12/31 becomes 1999/12/31 and 00/12/31 becomes 2000/12/31.

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the CENTURY command option or the CENTURY field specification parameter.

**CHEKPEND=(YES,)**

The CHEKPEND option specifies whether you want to set dependent table spaces to CHECK pending (CHKP) status. For details about CHKP status, see “Referential and check constraints” on page 146 and “CHECK CONSTRAINTS” on page 266.

The following table describes the action that LOADPLUS takes for each set of values that you can specify:
Table 123: CHEKPEND option values

<table>
<thead>
<tr>
<th>Values</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>(YES,)</td>
<td>Sets CHKP status unless overridden by CHECKPEND NO on the LOAD command</td>
</tr>
<tr>
<td>(NO,)</td>
<td>Does not set CHKP status unless overridden by CHECKPEND YES on the LOAD command</td>
</tr>
<tr>
<td>(YES,ENFORCE)</td>
<td>Sets CHKP status and allows only CHECKPEND YES on the LOAD command If CHECKPEND NO is specified on the LOAD command, LOADPLUS issues message BMC50115E and terminates.</td>
</tr>
<tr>
<td>(NO,ENFORCE)</td>
<td>Does not set CHKP status and allows only CHECKPEND NO on the LOAD command If CHECKPEND YES is specified on the LOAD command, LOADPLUS issues message BMC50115E and terminates.</td>
</tr>
</tbody>
</table>

When invoking DSNUTILB, LOADPLUS ignores this option.

*Overriding this option*

Unless you specify the ENFORCE keyword with this option, you can override the value for this option by using the CHECKPEND command option.

**COPYDDN=(BMCCPY, BMCCPZ)**

The COPYDDN option specifies default ddnames or ddname prefixes for the local copy data sets that receive either an image copy or a DSN1COPY-type copy of the table space or partitions that you are loading. If you are registering the copies, the first name is the local primary copy and the second name is the local backup copy.

When using this name as a prefix, allow sufficient bytes for the partition number to be added and still have a valid ddname (eight bytes or less). If you change the default ddnames from BMCCPY and BMCCPZ and you are not dynamically allocating your copy data sets, you must change the ddnames in your JCL. For more information, see “Copy data sets in LOADPLUS” on page 436.

If you are loading a partition-by-growth table space, ensure that the ddname prefix that you specify allows for the number of partitions specified for MAXPARTITIONS in your table space definition.

When invoking DSNUTILB, LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

This option does not apply to LOB or XML copy data sets.

*Overriding this option*
You can override the value for this option by using the COPYDDN command option.

**COPYLVL=FULL**

The COPYLVL option specifies how LOADPLUS is to assign dynamically allocated image copy data sets when loading a partitioned table space. You can specify one of the following options:

- **FULL** allocates a single copy data set to contain all partitions that are being loaded in either of the following cases:
  - You are loading all partitions.
  - You are loading a subset of contiguous partitions (specified on the PART command option), and you specify YES for the COPYSUBSET installation option.

- **PART** allocates individual copy data sets for each partition that you are loading. The following considerations apply to PART:
  - If you specify PART, LOADPLUS appends the partition number to the ddname prefixes that you specify in the COPYDDN and RCVYDDN options. The values that you specify for these options plus the highest partition number must not exceed eight characters. For more information, see “Specifying ddname prefixes” on page 81.
  - PART is not valid when LOADPLUS invokes DSNUTILB (and you specify COPY YES). If this value is in effect at runtime, LOADPLUS terminates.

**Overriding this option**

You can override the value for this option by using the COPYLVL command option. For additional considerations, see the command option description.

**COPYPEND=(YES,)**

The COPYPEND option specifies whether you want LOADPLUS to set the table space to COPY pending status if a copy was not made as part of the load job. The following table describes the actions that LOADPLUS takes for each set of values that you can specify:

<table>
<thead>
<tr>
<th>Values</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>(YES,)</td>
<td>Sets COPY pending status unless overridden by COPYPEND NO or NOCOPYPEND on the LOAD command</td>
</tr>
</tbody>
</table>
### Values and Action

<table>
<thead>
<tr>
<th>Values</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>(NO,)</td>
<td>Does not set COPY pending status unless overridden by COPYPEND YES on the LOAD command. For a DSNUTILB load, LOADPLUS passes this value to the IBM DB2 LOAD utility as NOCOPYPEND.</td>
</tr>
<tr>
<td>(YES,ENFORCE)</td>
<td>Sets COPY pending status and allows only COPYPEND YES on the LOAD command. If COPYPEND NO or NOCOPYPEND is specified on the LOAD command, LOADPLUS issues message BMC50115E and terminates.</td>
</tr>
<tr>
<td>(NO,ENFORCE)</td>
<td>Does not set COPY pending status and allows only COPYPEND NO on the LOAD command. If COPYPEND YES is specified on the LOAD command, LOADPLUS issues message BMC50115E and terminates. For a DSNUTILB load, LOADPLUS passes this value to the DB2 LOAD utility as NOCOPYPEND.</td>
</tr>
</tbody>
</table>

#### Overriding this option

Unless you specify the ENFORCE keyword with this option, you can override this value by using the COPYPEND command option.

### COPYSUBSET=NO

The COPYSUBSET option tells LOADPLUS whether to use a single copy data set for a subset of partitions. You can specify one of the following options:

- **NO** does not allow this single copy data set.
- **YES** enables a single copy data set. You must also meet the following requirements:
  - Specify a single subset of contiguous partitions with the PART command option.
  - For the local primary image copy, either activate dynamic allocation or allocate a single copy data set in your JCL.
  - If you dynamically allocate your copy data sets, specify FULL for the COPYLVL installation or command option.

### DELFILES=(YES,NO)

The DELFILES option tells LOADPLUS whether to delete the SORTOUT, SORTWK, SYSUT1, SYSERR, and SYSDISC files after the load completes successfully. The first value tells LOADPLUS whether to delete your SORTOUT, SORTWK, SYSUT1, and
SYSERR files after the load completes successfully, regardless of whether
LOADPLUS uses the files. The second value tells LOADPLUS how to handle the
discard file (SYSDISC). LOADPLUS deletes any SYSDISC files only if there were no
discarded records during the load process.

If you specify YES for these parameters, LOADPLUS deletes the appropriate file,
regardless of the JCL disposition for the data set.

The following table describes the action that LOADPLUS takes for each set of values
that you can specify:

Table 125: DELFILES option values

<table>
<thead>
<tr>
<th>Values</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Does not delete any work files</td>
</tr>
<tr>
<td>(NO,NO)</td>
<td>Does not delete any work files</td>
</tr>
<tr>
<td>(NO,YES)</td>
<td>Deletes only SYSDISC, if it is empty</td>
</tr>
<tr>
<td>YES</td>
<td>Deletes all work files except SYSDISC (assumes NO for the second value) regardless of the JCL disposition of these files and whether the files were used</td>
</tr>
<tr>
<td>(YES,YES)</td>
<td>Deletes all work files and SYSDISC (deletes SYSDISC only if it is empty) regardless of the JCL disposition of these files and whether the files were used</td>
</tr>
<tr>
<td>(YES,NO)</td>
<td>Deletes all work files except SYSDISC</td>
</tr>
</tbody>
</table>

Additional considerations

The following considerations apply to the DELFILES option:

- If the SYSDISC or SYSERR data set is allocated as a GDG data set, LOADPLUS
  deletes only the generation that was created during the current job.

- If your work files are defined with DISP=(any,DELETE,DELETE), LOADPLUS
  deletes these work files after the load completes even if you specify NO on this
  option.

- When LOADPLUS invokes DSNUTILB, SYSERR and SYSDISC files are not
  deleted, regardless of the value of this option.

- If you also specify a value for the RETPD or EXPDT option for your dynamically
  allocated SYSERR or SYSDISC data sets, LOADPLUS deletes your data sets only if
  the expiration date is earlier than the current date, or if RETPD is 0 or blank.

Overriding this option

You can override the value for this option by using the DELETEFILES and SYSDISC
command options.
Running in a worklist environment

If you are running the load job in a worklist environment, LOADPLUS functions differently. In this environment, LOADPLUS always ignores the value for the first parameter of the DELFILES installation option and, by default, does not delete the work files. LOADPLUS functions this way so that the work files exist for subsequent executions in the worklist. If you want to delete the work files, you must specify DETELEFILES YES on the LOAD command in the worklist.

For SYSDISC files in a worklist environment, LOADPLUS functions in one of the following ways:

- If you allocate your own SYSDISC file, LOADPLUS also ignores the value of the second parameter of the DELFILES installation option. In this case, if you want to delete SYSDISC (if it is empty), you must specify SYSDISC YES on the LOAD command in the worklist.

- If you have LOADPLUS dynamically allocate SYSDISC, LOADPLUS honors the value of the second parameter of the DELFILES installation option and deletes SYSDISC if that value is YES (and if SYSDISC is empty).

DISCARDLIMRC=8

The DISCARDLIMRC option allows you to specify a default user-defined return code that LOADPLUS uses when the number of discard violations reaches the discard limit that you specify with the DISCARDS command option. You can specify any integer greater than 7.

LOADPLUS always ends with the highest return code assigned during the load job. For more information, see “Return code hierarchy in LOADPLUS” on page 158.

When invoking DSNUTILB, LOADPLUS ignores this option.

Overriding this option

You can override the value for this option by using the IFDISCARDLIMIT command option.

DISCARDRC=0

The DISCARDRC option allows you to specify a default user-defined return code that LOADPLUS uses upon successful completion of the job when LOADPLUS discards one or more input records. You can specify any integer from 0 through 7.

LOADPLUS always ends with the highest return code assigned during the load job. For more information, see “Return code hierarchy in LOADPLUS” on page 158.
When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the IFDISCARDS command option.

**DISCDDN=SYSDISC**

The DISCDDN option specifies a default ddname for the discard data set. LOADPLUS places a record in this data set when that record cannot be loaded for any of the reasons that are listed in “SYSDISC data set in LOADPLUS” on page 450. If you change the default ddname from SYSDISC, you must also change the ddname in your JCL.

When invoking DSNUTILB, LOADPLUS uses this option only to pass a value for the &DDNAME variable for the data set name pattern. LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

**Overriding this option**

You can override the value for this option by using the DISCARDDN command option.

**DRNDELAY=1**

After a drain times out, the DRNDELAY option specifies the minimum number of seconds that you want LOADPLUS to wait before it tries again to obtain the drain. The number of seconds can range from 1 through 1800.

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the RETRY_DELAY command option.

**DRNRETRY=255**

The DRNRETRY option specifies the maximum number of times that you want LOADPLUS to attempt to obtain a drain before terminating the job. The number of retry attempts can range from 0 through 255.

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**
You can override the value for this option by using the RETRY command option.

**DRNWAIT=NONE**

The DRNWAIT option specifies the drain timeout value to use. Specify one of the following values:

- **NONE**, which tells the drain request issued by LOADPLUS to time out immediately if the drain cannot acquire the lock. NONE prevents any application transactions from being queued during the drain process. BMC recommends that you specify NONE in high-transaction environments.

- **UTIL**, which tells LOADPLUS to use the standard DB2 utility timeout value defined in DSNZPARMS for your site (IRLMRWT multiplied by UTIMOUT). The wait time applies to each object involved in the load.

- **SQL**, which tells LOADPLUS to use the standard SQL timeout value (IRLMRWT) as the drain timeout value. The wait time applies to each object involved in the load.

- Any integer value from 0 through 1800
  
  — 0 is equivalent to the value UTIL.
  
  — 1 through 1800 specifies the number of seconds to wait to obtain the drain for each drain retry before timing out.

If LOADPLUS cannot drain all of the objects within the time period specified by DRNWAIT, LOADPLUS completes the following process:

1. Releases the drains that it has obtained so far

2. Waits the length of time that you specify in the DRNDELAY installation option (or RETRY_DELAY command option)

3. Tries again to drain the objects for the number of times that you specify in the DRNRETRY installation option (or RETRY command option)

**Note**

The FORCE and FORCE_AT options tell LOADPLUS whether to cancel DB2 threads that might prevent a successful drain and, if so, when to cancel them. The value that you specify for the FORCE and FORCE_AT options might affect the drain process described here.

When invoking DSNUTILB, LOADPLUS ignores this option.

*Overriding this option*
You can override the value for this option by using the DRAIN_WAIT command option.

**DSNUEXIT=(NONE,ASM)**

The DSNUEXIT option specifies the name of a user-defined exit routine that creates data set name patterns, and the programming language in which the routine is written. This user exit provides the user-defined variables for constructing data set name patterns (with the DSNPAT option) that LOADPLUS uses when dynamically allocating data sets. For information about user-defined exit routines, see “LOADPLUS user exits” on page 825.

The following values are valid for the programming language:

- ASM
- COBOL2
- LE_COBOL
- C
- LE_C

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the DSNUEXIT command option.

**DSNUTILB=YES**

The DSNUTILB option tells LOADPLUS whether to invoke DSNUTILB to pass processing to the IBM DB2 LOAD utility. LOADPLUS uses this option to enable support for certain objects or features. For the list of features, see “Load jobs that invoke DSNUTILB” on page 57.

*Note*

LOADPLUS invokes DSNUTILB to enable new features quickly. LOADPLUS generally provides native support for these features (without invoking DSNUTILB) in a later version of the product or via PTF. You cannot use the DSNUTILB option (or any other option) to request that LOADPLUS invoke DSNUTILB for a feature that LOADPLUS supports natively.

You can specify one of the following values:

- YES tells LOADPLUS to invoke DSNUTILB when it is required to support the object type that is involved in the load. If you are loading a multi-table table space and any table in that table space (regardless of whether it is participating in the
LOAD (load) has a feature that is supported only by invoking DSNUTILB. LOADPLUS invokes DSNUTILB for the load job.

For requirements and restrictions that apply when LOADPLUS invokes DSNUTILB, see “Load jobs that invoke DSNUTILB” on page 57.

- NO tells LOADPLUS to not invoke DSNUTILB. If the type of load job that you are running requires DSNUTILB, LOADPLUS terminates.

**Overriding this option**

You can override the value for this option by using the DSNUTILB command option.

**ENFORCE=CHECK**

The ENFORCE option allows you to specify a default behavior for checking DB2-defined referential constraints and check constraints in your load jobs. The following table describes the options that you can specify:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONSTRAINTS</td>
<td>Tells LOADPLUS to check for violations of both referential constraints and check constraints during LOADPLUS processing</td>
</tr>
<tr>
<td></td>
<td>This option does not tell LOADPLUS to enforce informational referential constraints. Use the INFORI option to tell LOADPLUS how to handle informational referential constraints.</td>
</tr>
<tr>
<td>NO</td>
<td>Tells LOADPLUS to not check for violations of either referential constraints or check constraints during LOADPLUS processing</td>
</tr>
<tr>
<td></td>
<td>When LOADPLUS invokes DSNUTILB, the following actions occur:</td>
</tr>
<tr>
<td></td>
<td>- For an SQLAPPLY load, LOADPLUS passes ENFORCE CONSTRAINTS.</td>
</tr>
<tr>
<td></td>
<td>- Otherwise, LOADPLUS passes ENFORCE NO.</td>
</tr>
<tr>
<td>RI</td>
<td>Tells LOADPLUS to check for violations of referential constraints only</td>
</tr>
<tr>
<td></td>
<td>This option does not tell LOADPLUS to enforce informational referential constraints. Use the INFORI option to tell LOADPLUS how to handle informational referential constraints.</td>
</tr>
<tr>
<td></td>
<td>When LOADPLUS invokes DSNUTILB, the following actions occur:</td>
</tr>
<tr>
<td></td>
<td>- For an SQLAPPLY load, LOADPLUS passes ENFORCE CONSTRAINTS.</td>
</tr>
<tr>
<td></td>
<td>- Otherwise, LOADPLUS issues message BMC50178E and terminates.</td>
</tr>
<tr>
<td>CHECK</td>
<td><em>(default)</em> Tells LOADPLUS to check for violations of check constraints only</td>
</tr>
<tr>
<td></td>
<td>When LOADPLUS invokes DSNUTILB, the following actions occur:</td>
</tr>
<tr>
<td></td>
<td>- For an SQLAPPLY load, LOADPLUS passes ENFORCE CONSTRAINTS.</td>
</tr>
<tr>
<td></td>
<td>- Otherwise, LOADPLUS issues message BMC50178E and terminates.</td>
</tr>
</tbody>
</table>
For more information about how LOADPLUS handles referential and check constraints, see “Referential and check constraints” on page 146.

**Overriding this option**

You can override the value for this option by using the ENFORCE command option.

**ERRDDN=SYSERR**

The ERRDDN option specifies a default ddname for the error data set (the data set that contains information about records that are discarded because of errors). If you change the default ddname from SYSERR, you must also change the ddname in your JCL. For more information, see “SYSERR data set in LOADPLUS” on page 453.

When invoking DSNUTILB, LOADPLUS uses this option only to pass a value for the &DDNAME variable for the data set name pattern. LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

**Overriding this option**

You can override the value for this option by using the ERRDDN command option.

**EXCLDUMP=(X37,X22,X06)**

The EXCLDUMP option allows you to limit the conditions under which LOADPLUS generates a system dump when the SDUMP option contains values other than NO. This option tells LOADPLUS to exclude the listed abend codes when generating the dump.

Specify a three-digit abend code, such as 806 or 222, or prefix an 'X' to a two-digit abend code to exclude all abend codes that end in those two digits. For example, if you specify X37, LOADPLUS will not generate a system dump for all abend codes that end in 37 (such as B37 or E37). You can specify up to 10 abend codes.

Specify EXCLDUMP=0 (without parentheses) if you want all abend codes to be candidates for a system dump.

**FASTSWITCH=NO**

*This option is valid only for SHRLEVEL CHANGE (except an SQLAPPLY load) or SHRLEVEL REFERENCE load jobs.*

The FASTSWITCH option tells LOADPLUS how to handle staging data sets after the load process is complete. Valid values are NO, YES, and ZPARM:
• NO tells LOADPLUS to rename the staging data sets to the original data set names.

• YES tells LOADPLUS to bypass the VSAM rename process and directly update the DB2 catalog to point to the staging data sets.

Consider the following information about the FASTSWITCH process:

— When loading clone tables or base objects that participate (or have participated) in a clone relationship, LOADPLUS changes FASTSWITCH YES to FASTSWITCH NO.

— BMC recommends using the FASTSWITCH process when you are loading objects with more than 200 data sets, such as a segmented table space that contains many tables that each have an index.

— The FASTSWITCH process requires updates to the DB2 catalog and directory. In some environments, LOADPLUS might have difficulty acquiring the necessary locks, which results in contention or deadlocks. Minimizing use of the FASTSWITCH process can reduce this contention and allow the process to be more effective when it is needed. Frequently reorganizing the DB2 catalog and directory can also help prevent contention.

• ZPARM tells LOADPLUS to use the value that is set in IBM DSNZPARM for FASTSWITCH. FASTSWITCH=ZPARM, enables you to use DB2 parameters to manage the FASTSWITCH setting, without changing your LOADPLUS installation options module.

For more information about these methods for handling staging data sets, see “Staging data sets” on page 98.

**Overriding this option**

You can override the value for this option by using the FASTSWITCH command option.

**FILECHK=FAIL**

The FILECHK option specifies the action that LOADPLUS should take when encountering a work file that has been allocated in one of the following ways:

• As a temporary data set

  For a definition of temporary data sets within LOADPLUS, see “Check for data set attributes” on page 85.

• As DD DUMMY or DSN=NULLFILE

Specify one of the following options:
- FILECHK=FAIL tells LOADPLUS to terminate.
- FILECHK=WARN tells LOADPLUS to issue a warning message and continue processing.

When invoking DSNUTILB, LOADPLUS ignores this option.

**FILEVAR (obsolete)**

In version 11.1 and earlier, this option told LOADPLUS whether to allow file reference subtypes in your field specifications when loading LOB or XML data via DSNUTILB.

Beginning with version 11.2, LOADPLUS no longer requires or supports the FILEVAR option. If your installation options module contains this option, the module compiles successfully but completes with return code 4 and a message indicating that this option is obsolete.

**FORCE=None**

The FORCE option specifies whether to cancel DB2 threads that might prevent a drain process from completing. Specify one of the following options:

- NONE tells LOADPLUS not to cancel DB2 threads.
- READERS tells LOADPLUS to cancel read claimers at the point specified by the FORCE_AT option.
- ALL tells LOADPLUS to cancel both read and write claimers at the point specified by the FORCE_AT option.

When invoking DSNUTILB, LOADPLUS ignores this option.

*Overriding this option*

You can override the value for this option by using the FORCE command option. The FORCE command option provides an additional option, REPORTONLY, that tells LOADPLUS to display a report of the threads that would have been canceled if you had specified FORCE ALL.

**FORCE_AT=(START,3)**

The FORCE_AT option tells LOADPLUS when to cancel DB2 threads that might prevent the drain process from completing. This option is applicable only when FORCE READERS, FORCE ALL, or FORCE REPORTONLY is in effect.
The first parameter of this option indicates at which point during the drain process to begin canceling threads. The second parameter indicates how long to delay the start of the cancelation process from the specified point in the drain process.

For the first parameter of this option, specify one of the following values:

- **START** tells LOADPLUS to start canceling threads when the drain request begins.
- **RETRY** tells LOADPLUS to start canceling threads the first time the drain process times out and LOADPLUS attempts to retry the drain.
- **LASTRETRY** tells LOADPLUS to start canceling threads at the beginning of the last retry attempt following a drain process timeout. You can control the number of drain retry attempts with the RETRY command option or DRNRETRY installation option.

**Note**
The cancelation process includes a small amount of time to issue the cancel commands, plus any time that DB2 needs to roll back the canceled transactions. If the drain wait time (controlled by the DRNWAIT option) is shorter than the total cancelation process time, the drain fails. The START and RETRY values allow for additional drain attempts after the start of the cancelation process; therefore, these values increase the likelihood that the drain will succeed when a short drain wait time is in effect. (Alternatively, you can increase the drain wait time.)

For the second parameter of this option, specify an integer value to indicate hundredths of a second. For example, specify 7 to tell LOADPLUS to wait .07 seconds. Specify an integer value of 0 or greater. The default, 0, tells LOADPLUS to start the cancelation process immediately upon reaching the point specified by the FORCE_AT option.

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**
You can override the value for the first parameter of this option by using the FORCE_AT suboption of the FORCE command option. You can override the value for the second parameter by using the FORCE_DELAY suboption of the FORCE command option.

**FORCE_RPT=NO**
The FORCE_RPT option tells LOADPLUS whether to display a report of the canceled threads.

Specify one of the following values for this option:
YES tells LOADPLUS to display the report. This value is applicable only when
FORCE READERS or FORCE ALL is in effect.

If your JCL includes a BMCFORCE DD statement, LOADPLUS sends the canceled
threads report to that data set. Otherwise, the report is displayed in the
LOADPLUS SYSPRINT.

NO tells LOADPLUS not to display the report.

LOADPLUS ignores this option when either of the following conditions exists:

- LOADPLUS is invoking DSNUTILB
- You specify FORCE REPORTONLY on the LOAD command

**Overriding this option**

You can override the value for this option by using the FORCE_RPT suboption of the
FORCE command option.

**HASHAX (obsolete)**

In version 11.1 and earlier, this option told LOADPLUS whether to load (by
invoking DSNUTILB) a table that was defined as ORGANIZE BY HASH.

Beginning with version 11.2, LOADPLUS no longer requires or supports the
HASHAX installation option. If your installation options module contains this
option, the module compiles successfully but completes with return code 4 and a
message indicating that this option is obsolete.

**IBUFFS=20**

The IBUFFS option specifies a multiple of either 32 KB or 256 KB to use to define the
amount of buffer pool storage for each active input (SYSREC) data set. LOADPLUS
allocates as many buffers as will fit into the calculated space. The multiplier that
LOADPLUS applies to the IBUFFS option value varies as follows:

- **For SYSREC data sets with a block size greater than 32 KB**, LOADPLUS uses a
  multiplier of 256 KB. In this case, if the IBUFFS option has a value greater than 6,
  LOADPLUS ignores that setting and uses 6.

- **For all other SYSREC data sets**, LOADPLUS uses a multiplier of 32 KB.

For more information, see “Performance considerations for buffer options” on page
656.
**IDCACHE=1000**

The IDCACHE option allows you to specify the size of the cache that LOADPLUS uses when generating values for an identity column. Specify the size as the number of values to reserve. LOADPLUS reserves this cache in SYSIBM.SYSSEQUENCES and reserves at least one cache for each read task. Valid values for this option are 1 through 100000.

*Note*

This cache is a temporary cache that LOADPLUS uses and does not affect the cache that you specified when defining your identity column.

When invoking DSNUTILB, LOADPLUS ignores this option.

*Overriding this option*

You can override the value for this option by using the IDCACHE command option.

**IDCDDN=SYSIDCIN**

The IDCDDN option specifies a default ddname for the input data set containing the IDCAMS command statements that LOADPLUS uses to redefine VSAM data sets. For more information, see “SYSIDCIN data set in LOADPLUS” on page 462.

*Overriding this option*

You can override the value for this option by using the IDCDDN command option.

**IDERROR=DISCARD**

The IDERROR option tells LOADPLUS how to handle rows with generated identity columns values that are outside the range that is defined on the identity column. Valid values are DISCARD and FAIL:

- DISCARD tells LOADPLUS to discard these rows. DISCARD is not valid when LOADPLUS invokes DSNUTILB. If this value is in effect at runtime for a DSNUTILB load, LOADPLUS issues message BMC50178E and terminates.
- FAIL tells LOADPLUS to terminate as soon as it encounters a generated identity column value that is outside this range. LOADPLUS ignores this option for an SQLAPPLY load job.

*Overriding this option*

You can override the value for this option by using the IDERROR command option.
INDDN=SYSREC

The INDDN option specifies a default ddname or ddname prefix for the input data set. When using this name as a prefix, allow sufficient bytes for LOADPLUS to add the number of input files and still have a valid ddname (eight bytes or less). If you change the default ddname from SYSREC and you are not using the INDSN command option, you must change the ddname in your JCL. For more information, see “SYSREC data sets in LOADPLUS” on page 456.

If any DD statements in your JCL match the value for INDDN and you specify the INDSN command option, LOADPLUS issues message BMC50460E and fails.

Overriding this option

You can override the value for this option by using the INDDN command option.

INFORI=NO

The INFORI option tells LOADPLUS how to handle the enforcement of informational referential constraints:

- NO tells LOADPLUS to ignore informational referential constraints.
- YES tells LOADPLUS to enforce informational referential constraints in the same way as other referential constraints.

Be aware of the following information when you specify INFORI=YES:

- LOADPLUS discards any violating rows, just as it does for other referential constraints.

- If LOADPLUS sets CHKP status on a table because it enforced informational referential constraints on the parent table, you might not be able to use the standard methods for resetting CHKP status. Therefore, if you are enforcing only informational constraints, BMC recommends that you also specify CHECKPEND NO.

  For more information about resetting CHKP status, see “Restoring integrity” on page 151.

- When LOADPLUS invokes DSNUTILB, the following actions occur:
  - For an SQLAPPLY load, LOADPLUS ignores this option.
  - Otherwise, if this value is in effect at runtime, LOADPLUS issues message BMC50178E and terminates.

Overriding this option
You can override the value for this option by using the INFORI option of the ENFORCE CONSTRAINTS or ENFORCE RI command options.

**INLINECP=YES**

The INLINECP option tells LOADPLUS whether to create inline image copies. You can specify YES or NO:

- **YES** tells LOADPLUS to create an inline image copy while loading the table space. This value requires that COPY YES be specified on the LOAD command.

- **NO** tells LOADPLUS to not make an inline image copy while loading the table space, but to create an image copy after loading the table space.

When invoking DSNUTILB, LOADPLUS ignores this option.

If any of the following conditions apply to your load job, LOADPLUS functions as if you specified NO, regardless of the value that you specify either here or on the LOAD command:

- You are loading XML data.

- You specify LOAD RESUME YES, unless you also specify PART \( n \) REPLACE for all participating partitions.

- The copy data sets are on a stacked tape.

- You restart your load job, with the following exception: If you are running a two-phase load job and you specify RESTART(PHASE) on your EXEC statement, LOADPLUS generates an inline copy if the value of INLINE is YES.

- The size of the table space page is greater than 4 KB, you are loading multiple partitions, and you have one image copy data set.
  
  An exception to this condition exists for partition-by-growth table spaces. LOADPLUS can create inline image copies of partition-by-growth table spaces in this case.

*Overriding this option*

You can override the value for this option by using the INLINE command option.

**INLOB (obsolete)**

In LOADPLUS versions 10.2 and earlier, this option told LOADPLUS whether to load (by invoking DSNUTILB) table spaces that contain an inline LOB column.
Beginning with version 11.1, LOADPLUS no longer requires this option. The value of the DSNUTILB option determines whether LOADPLUS will invoke DSNUTILB to load inline LOB data.

If your installation options module contains the INLOB option, the module compiles successfully but completes with return code 4 and a message indicating that this option is obsolete.

**IXONEX (obsolete)**

In LOADPLUS 11.1 and earlier, this option told LOADPLUS whether to load (by invoking DSNUTILB) table spaces that contained an index that contained a key derived from certain expressions.

Beginning with version 11.2, LOADPLUS no longer requires or supports the IXONEX installation option. If your installation options module contains this option, the module compiles successfully but completes with return code 4 and a message indicating that this option is obsolete.

**IXRANDOM (obsolete)**

In LOADPLUS 11.1 and earlier, this option told LOADPLUS whether to load (by invoking DSNUTILB) table spaces that contained an index that contained a key with random ordering.

Beginning with version 11.2, LOADPLUS no longer requires or supports the IXRANDOM installation option. If your installation options module contains this option, the module compiles successfully but completes with return code 4 and a message indicating that this option is obsolete.

**KEEPDICTIONARY=NO**

The KEEPDICTIONARY option tells LOADPLUS whether to keep the existing compression dictionary when running a LOAD REPLACE or LOAD RESUME YES PART n REPLACE job:

- **NO** tells LOADPLUS to build a new compression dictionary.
- **YES** tells LOADPLUS to keep the existing compression dictionary. If a dictionary does not exist, LOADPLUS builds one.

When LOADPLUS invokes DSNUTILB, the following actions occur:

- For an SQLAPPLY load, LOADPLUS ignores this option.
- Otherwise, LOADPLUS passes KEEPDICTIONARY=YES to the IBM DB2 LOAD utility as KEEPDICTIONARY, and ignores KEEPDICTIONARY=NO.
For details about when LOADPLUS builds and keeps a dictionary, see “Considerations when using table space compression” on page 140.

Restrictions

The following restrictions apply to the KEEPDICTIONARY option:

- The KEEPDICTIONARY option is valid only if the table space or partition that you are loading has the COMPRESS YES attribute.

- If a table space is compressed and a load job would convert the row format from BRF to RRF, LOADPLUS builds a new dictionary except when both of the following options are in effect:
  — The value of the LOADPLUS KEEPDICTIONARY option is YES.
  — The value of the DB2 subsystem parameter HONOR_KEEPDICTIONARY is YES.

Overriding this option

You can override the value for this option either at the table space level, by using the KEEPDICTIONARY option with LOAD REPLACE, or at the partition level, by using the KEEPDICTIONARY option with PART n REPLACE.

LBUFFS=20

The LBUFFS option specifies a multiple of 32 KB to use to define the amount of buffer pool storage that is allocated for each active load (SORTOUT) data set. LOADPLUS allocates as many buffers as will fit into the calculated space. For more information, see “Performance considerations for buffer options” on page 656.

LOADDN=SORTOUT

The LOADDN option specifies a default ddname or ddname prefix for the output data set from the PRELOAD phase. When using this name as a prefix for a partitioned table space, allow sufficient bytes for the partition number to be added and still have a valid ddname (eight bytes or less). If you change the default ddname from SORTOUT and you are not dynamically allocating your SORTOUT data sets, you must change the ddname in your JCL. For more information, see “SORTOUT data sets in LOADPLUS” on page 443.

Additional considerations

The following considerations apply to the LOADDN option:

- If you specify or default to a prefix with a length of 7 or more characters, and the value for the SMAX installation option is greater than 9, LOADPLUS reduces the value of SMAX to 9.
When invoking DSNUTILB, LOADPLUS uses this option only to pass a value for the &DDNAME variable for the data set name pattern. LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

**Overriding this option**

You can override the value for this option by using the LOADDN command option.

**LOB (obsolete)**

In LOADPLUS 11.1 and earlier, this option told LOADPLUS whether to load LOB data in a LOB table space when DSNUTILB was required.

Beginning with version 11.2, LOADPLUS no longer requires or supports the LOB installation option. If your installation options module contains this option, the module compiles successfully but completes with return code 4 and a message indicating that this option is obsolete.

**LOBAVGPCT=50**

The LOBAVGPCT option provides LOADPLUS with an estimate of the size of your LOB data. LOADPLUS uses this information to balance the number of tasks needed to load the base table, load the LOB table space, and build the indexes. LOADPLUS also uses this information when allocating LOB copy data sets.

Specify a percentage of the maximum size defined for the LOB that you are loading. Valid values are 1 through 100.

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option either at the global level, by using the LOBAVGPCT option, or at the field level, by using the AVGSIZE field option on the BLOB, BLOBF, CLOB, CLOBF, DBCLOB or DBCLOBF field specification. The field-level option overrides the global option.

**LOCKROW=YES**

The LOCKROW option tells LOADPLUS which serialization method to use when updating the BMCSYNC and BMCUTIL tables:

- YES tells LOADPLUS to use MVS enqueues instead of SQL LOCK TABLE statements for serialization. Using LOCKROW=YES should prevent most SQL -911 return codes that occur when multiple BMC products concurrently update these tables.
NO tells LOADPLUS to use SQL LOCK TABLE statements for serialization when updating the BMCSYNC and BMCUTIL tables.

Additional considerations

The following considerations apply to the LOCKROW option:

- Using LOCKROW=YES requires row-level locking. You must define the BMCSYNC and BMCUTIL tables with LOCKSIZE ROW to achieve row-level locking.

- The following BMC products must use the same LOCKROW value within a subsystem: CHECK PLUS, COPY PLUS, LOADPLUS, RECOVER PLUS, REORG PLUS, and UNLOAD PLUS. Regardless of the value assigned for LOCKROW, COPY PLUS and RECOVER PLUS always behave as if LOCKROW=YES.

- In a future version, LOADPLUS will no longer require or support the LOCKROW installation option.

**LONGNAMETRUNC=MIDDLE**

The LONGNAMETRUNC option tells LOADPLUS where to truncate names that are longer than the area that is available in a LOADPLUS report-style message. The following table describes the valid values and the symbol that appears in the message based on that value:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIDDLE (or M)</td>
<td>Truncates outward from the middle of the name</td>
<td>&gt;&gt;</td>
</tr>
<tr>
<td>BEGINNING (or B)</td>
<td>Truncates from the beginning of the name</td>
<td>&lt;&lt;</td>
</tr>
<tr>
<td>END (or E)</td>
<td>Truncates from the end of the name</td>
<td>&gt;&gt;</td>
</tr>
</tbody>
</table>

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the LONGNAMETRUNC command option.

**MAPDDN=SYSMAP**

The MAPDDN option allows you to specify the default &DDNAME variable for the data set name pattern for the mapping data set. LOADPLUS includes this
information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

**Overriding this option**

You can override the value for this option by using the MAPDDN command option.

**MAXP=5**

The MAXP option specifies the maximum number of partitions to load concurrently. For more information, see “Performance considerations for the MAXP installation option” on page 661.

**MAXSORTMEMORY=0**

The value that you specify, in kilobytes, for the MAXSORTMEMORY option helps LOADPLUS determine the maximum amount of memory to allocate to each sort task.

BMC strongly recommends that you use the shipped value of 0, which tells LOADPLUS to automatically compute the maximum amount of memory that is needed to perform each sort task. In addition to 0, you can specify any value between the value that you specify for the MINSORTMEMORY option and 2097152.

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the MAXSORTMEMORY command option.

**MAXTAPE=3**

The MAXTAPE option specifies the maximum number of tape drives to dynamically allocate at one time. The value must be greater than zero.

For partition-by-growth table spaces, if you are making partition-level copies and sending them to tape, ensure that the value for MAXTAPE takes potential expansion into account. For details about determining this value, see “MAXTAPE” on page 394.

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the MAXTAPE command option.
MGEXTENT=CONTINUE

The MGEXTENT option specifies how to allocate the extents of a DB2 object when creating a new data set during extend processing. Specify one of the following options:

- CONTINUE tells LOADPLUS to allocate the extents as follows:
  - Allocate the primary extent with the greater of the original primary quantity or the last secondary quantity of the previous data set.
  - Allocate the first secondary extent with the last secondary quantity of the previous data set.

- RESET tells LOADPLUS to allocate the primary and first secondary extents by using the original values from the DB2 object allocation.

With the exception of this option, LOADPLUS allocates secondary extents in the same way that DB2 does. LOADPLUS uses a sliding-scale calculation for secondary extents, if applicable, that is similar to the method that DB2 uses. For more information, see the DB2 documentation that describes the sliding-scale algorithm.

**Note**

The default for this new option changes the way that LOADPLUS functions from the way that it functioned in versions 9.3.00 and earlier. Review the values for this option to ensure that you obtain the results that you intend.

MINSORTMEMORY=0

The value, in kilobytes, that you specify for the MINSORTMEMORY option helps LOADPLUS determine the minimum amount of memory to allocate to each sort task.

BMC strongly recommends that you use the shipped value of 0, which tells LOADPLUS to automatically compute the minimum amount of memory that is needed to perform each sort task. In addition to 0, you can specify any value between 1280 and the value that you specify for the MAXSORTMEMORY installation or command option.

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the MINSORTMEMORY command option.
MSGLEVEL=1

The MSGLEVEL option controls the messages that LOADPLUS returns in the SYSPRINT and SYSPRINT2 data sets. MSGLEVEL=0 returns minimal messages. MSGLEVEL=1 returns additional messages to help you diagnose problems and fine-tune performance.

Overriding this option

You can override the value for this option by using the MSGLEVEL parameter on the EXEC statement.

OPNDB2ID=YES

The OPNDB2ID option tells LOADPLUS whether to use the DB2 RACF ID or the user’s RACF ID:

- YES tells LOADPLUS to use the DB2 RACF ID (instead of the RACF ID of the user who is running LOADPLUS) when opening or performing Access Method Services (AMS) functions on DB2 data sets.

- NO tells LOADPLUS to use the RACF ID of the user who is running LOADPLUS. If you specify NO, the user must have the appropriate RACF authorization.

For any security system other than RACF, specify OPNDB2ID=NO to have LOADPLUS use the security authorization ID of the user who is running LOADPLUS.

ORIGDISP=DELETE

This option is valid only for SHRLEVEL CHANGE (except an SQLAPPLY load) or SHRLEVEL REFERENCE load jobs.

The ORIGDISP option specifies the action that LOADPLUS should take after it has successfully renamed or switched the staging data sets during a SHRLEVEL CHANGE (except SQLAPPLY) or SHRLEVEL REFERENCE job:

- DELETE tells LOADPLUS to delete the original data sets.

- RENAME tells LOADPLUS to rename the original data sets to the staging data set names.

  RENAME preserves the space that was initially allocated for the original data sets. As a result, the staging data sets are ready to use in a subsequent load job.

  LOADPLUS ignores a value of RENAME if either of the following conditions exists:
— You specify YES for the FASTSWITCH option.

— You specify ZPARM for the FASTSWITCH installation option, and the DB2 ZPARM value is YES.

In these cases, no rename is needed. The data sets keep their original names, and the space is preserved for use in a subsequent load job.

For more information about the rename and FASTSWITCH processes, see “Staging data sets” on page 98.

**Overriding this option**

You can override the value for this option by using the ORIGINALDISP command option.

**PAUSEDISCARDRC=4**

The PAUSEDISCARDRC option allows you to designate a default user-defined return code that LOADPLUS uses when pausing after the PRELOAD phase due to discarded records. The value for this option is in effect only if you specify PRELOAD PAUSE ANYDISCARDS on the LOAD command.

You can specify any positive integer. Note the following results for certain values:

- LOADPLUS always ends with the highest return code assigned during the load job. Therefore, because PRELOAD PAUSE normally ends with return code 4, specifying a value of 4 or less has no effect. For more information, see “Return code hierarchy in LOADPLUS” on page 158.

- If you specify an integer that is greater than 31, LOADPLUS issues a user abend that is equal to the integer that you specify.

- If you specify an integer that is greater than 4095, the LOADPLUS job step terminates with a system abend 001.

**Overriding this option**

You can override the value for this option by using the RETCODE keyword of the PRELOAD PAUSE ANYDISCARDS command option.

**PLAN=AMU1120**

The PLAN option specifies the name of the product plan. This plan contains the packages that provide the various capabilities of LOADPLUS.
**PREFORMAT=NO**

The PREFORMAT option tells LOADPLUS whether to preformat unused data set pages:
- NO tells LOADPLUS not to preformat unused pages in a data set.
- YES tells LOADPLUS to write full pages initialized with zeros up to the high-allocated RBA of the participating table space and index spaces.

When invoking DSNUTILB, LOADPLUS passes PREFORMAT=YES to the IBM DB2 LOAD utility as PREFORMAT and ignores PREFORMAT=NO.

*Overriding this option*

You can override the value for this option, either at the table space level or at the partition level, by using the appropriate PREFORMAT option.

**PRELOAD=LOAD**

The PRELOAD option specifies whether to use two-phase or single-phase processing for your load jobs. You can specify CONTINUE or LOAD for the installation option.

PRELOAD=LOAD tells LOADPLUS to use single-phase load processing, which combines the functions of the PRELOAD and LOAD phases into the COMBINED phase. This value can improve performance in most cases.

PRELOAD=CONTINUE tells LOADPLUS to use two-phase processing and to continue with the LOAD phase after the PRELOAD phase is completed.

*Considerations*

The following considerations apply to the PRELOAD option:
- If PRELOAD CONTINUE is in effect at runtime when you are loading LOB or XML data, LOADPLUS invokes DSNUTILB.
- LOADPLUS terminates if PRELOAD CONTINUE is in effect at runtime and you also specify ENFORCE RI or ENFORCE CONSTRAINTS, LOADPLUS terminates.
- LOADPLUS ignores the value for this option in the following cases:
  - When invoking DSNUTILB
  - When you are running an SQLAPPLY load

SQLAPPLY load jobs always use single-phase processing.

*Overriding this option*
You can override the value for this option by using the PRELOAD command option. Additional values are available with the command option.

**RCVYDDN=(BMCRCY, BMCRCZ)**

The RCVYDDN option specifies default ddnames or ddname prefixes for remote copy data sets. These data sets are the ones that receive either an image copy or a DSN1COPY-type copy of the table space or partitions that you are loading. If you are registering the copies, the first name is the remote primary copy, and the second name is the remote backup copy.

When using this name as a prefix, ensure that you allow sufficient bytes for the partition number to be added and still have a valid ddname (eight bytes or less). In addition, if you change the default ddnames from BMCRCY and BMCRCZ and you are not dynamically allocating your copy data sets, you must change the names in your JCL. For more information, see “Copy data sets in LOADPLUS” on page 436.

If you are loading a partition-by-growth table space, ensure that the ddname prefix that you specify allows for the number of partitions specified for MAXPARTITIONS in your table space definition.

When invoking DSNUTILB, LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

This option does not apply to LOB or XML copy data sets.

**Overriding this option**

You can override the value for this option by using the RECOVERYDDN command option.

**REDEFINE= YES**

The REDEFINE option tells LOADPLUS whether to delete and redefine the VSAM data sets for the table space and index spaces that are participating in the load. LOADPLUS can delete and redefine both user-defined (VCAT-defined) data sets and data sets that are defined in DB2 storage groups (STOGROUP-defined).

You can specify the following values:

- For SHRLEVEL NONE, REDEFINE= YES tells LOADPLUS to delete and redefine the VSAM data sets for the space before loading it. For jobs that use staging data sets, LOADPLUS deletes and defines the staging data sets.
- REDEFINE= NO tells LOADPLUS not to delete and redefine the existing VSAM data sets for the table space or indexes. Instead, LOADPLUS issues message BMC50391I, reuses the existing data sets, and resets the high-used RBA.
**Additional considerations**

The following considerations apply to the REDEFINE option:

- If you anticipate loading a large number of partitions, BMC recommends that you specify REDEFINE NO.

- When invoking DSNUTILB, LOADPLUS passes REDEFINE=NO to the IBM DB2 LOAD utility as REUSE and ignores REDEFINE=YES.

**Overriding this option**

You can override the value for this option by using the REDEFINE command option. For information about the command option and more information about how LOADPLUS functions under each value of this option, see “REDEFINE” on page 246.

**RENMMAEX=30**

The RENMMAEX option indicates the maximum number of tasks to start during the staging data set rename and delete process. You can specify the following values:

- 0 to have LOADPLUS determine the number of tasks to start

- \( n \) to specify that LOADPLUS can start a maximum of \( n \) tasks (where \( n \) is a positive integer from 1 through 32767)

**RULES=STANDARD**

The RULES option specifies the set of comparison rules for LOADPLUS to use in WHEN, NULLIF, and DEFAULTIF processing. You can specify STANDARD or BMC. This specification determines the following information:

- The order in which WHEN testing and data conversion, if any, is performed

- Which set of operators can be used in comparisons

- How constants are handled during translation

For information about the differences between STANDARD and BMC, see “RULES installation option” on page 819.

When your LOAD command contains the FORMAT UNLOAD, FORMAT BMC, or FORMAT BMCCUNLOAD option, LOADPLUS changes the value of the RULES option to BMC, regardless of the value that you specify. When your LOAD command contains the FORMAT CSV option, LOADPLUS changes the value of this option to STANDARD, regardless of the value that you specify.
**SDUMP=(ALLPSA, CSA, RGN, SQA, LSQA, SUM, TRT, IO)**

The SDUMP option tells LOADPLUS to generate a system dump, using the information listed in the option, if a load job abnormally terminates. The values listed provide diagnostic information to BMC Customer Support. Although you can specify any values that IBM allows for the SDATA parameter on the SDUMPX macro, BMC recommends that you do not change the values that were shipped with the product. For a complete list and description of the values that you can specify for this option, see the IBM SDUMPX macro description.

LOADPLUS uses the system-defined dump data set to hold the data. In cases where multiple abends occur, LOADPLUS generates the dump for only the first abend. The SDUMP option allows you to generate a system dump, regardless of your access to storage keys 0 through 7.

You can specify SDUMP=NO (without parentheses) to not generate a system dump.

**Note**

LOADPLUS displays SDUMP=YES in message BMC50471I in the SYSPRINT list of options when SDUMP contains one or more values.

To limit the conditions under which LOADPLUS generates the system dump, you can exclude selected abend codes by using the EXCLDUMP option.

**SHORTMEMORY=CONTINUE**

The SHORTMEMORY option controls the action that LOADPLUS should take when one of the following memory shortages exists during sort processing:

- The system contains insufficient available or total pages of memory for LOADPLUS to perform an optimal sort.
- The region contains insufficient memory for LOADPLUS to perform a minimum number of tasks, given the amount of memory required to perform an optimal sort. However, the region does contain at least 1280 KB of memory, or the minimum amount of memory specified by the MINSORTMEMORY option. If the region contains 1280 KB of memory and the amount of memory specified by MINSORTMEMORY is available, LOADPLUS uses the greater amount of memory.

You can specify one of the following values for the SHORTMEMORY option:

- CONTINUE, the shipped value, indicates that, when a memory shortage exists, LOADPLUS should continue sort processing.
- FAIL indicates that LOADPLUS should fail when a memory shortage exists.
When invoking DSNUTILB, LOADPLUS ignores this option.

*Overriding this option*

You can override the value for this option by using the SHORTMEMORY command option.

**SHRLEVEL=(NONE,NONE)**

The SHRLEVEL option specifies the level of access that DB2 has, during LOADPLUS processing, to the objects that you are loading. The first parameter provides the default SHRLEVEL value for LOAD RESUME NO and LOAD REPLACE jobs. The second parameter provides the default SHRLEVEL value for LOAD RESUME YES jobs.

You can specify the values that the following table describes:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Parameter for which value is valid</th>
<th>Additional considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>This value tells LOADPLUS to stop the objects that you are loading, making them unavailable during the entire load process</td>
<td>Both</td>
<td>None</td>
</tr>
<tr>
<td>REFERENCE</td>
<td>This value tells LOADPLUS to make the objects that you are loading available in read-only status. LOADPLUS writes the loaded data to staging data sets. The original VSAM data sets remain intact throughout the load process, allowing you to easily restart from a failure or make the objects available without having to recover.</td>
<td>Both</td>
<td>This value is not valid when LOADPLUS invokes DSNUTILB. If this value is in effect at runtime, LOADPLUS issues message BMC50178E and terminates.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
<td>Parameter for which value is valid</td>
<td>Additional considerations</td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CHANGE</td>
<td>This value tells LOADPLUS to allow the objects that you are loading to remain in read/write status while LOADPLUS writes the loaded data to staging data sets. The original VSAM data sets remain intact throughout the load process. This nondestructive process allows you to easily restart from a failure or make the objects available without having to recover.</td>
<td>Both</td>
<td>For the second parameter (LOAD RESUME YES), this value applies only to non-SQLAPPLY load jobs when you specify PART n REPLACE. If you do not specify PART n REPLACE and this value is in effect, LOADPLUS uses SQLAPPLY. This option is not valid when LOADPLUS invokes DSNUTILB. If this value is in effect at runtime, LOADPLUS issues message BMC50178E and terminates.</td>
</tr>
<tr>
<td>SQLAPPLY</td>
<td>This value tells LOADPLUS to allow the object that you are loading to remain in read/write status while LOADPLUS loads the data by using SQL insert processing. For this type of load, LOADPLUS uses the High-speed Apply Engine component of the BMC Log Master for DB2 product. This feature does not require that you have a license for the Log Master product. However, the High-speed Apply Engine must be installed and made available through the STEPLIB, JOBLIB, or LINKLIST at runtime.</td>
<td>Second</td>
<td>This value is valid only for LOAD RESUME YES jobs.</td>
</tr>
</tbody>
</table>

**Note**

You can also specify a single value, indicating that the other parameter should be NONE, as shown in the following examples:

- SHRLEVEL=REFERENCE indicates that the default is SHRLEVEL REFERENCE for LOAD REPLACE jobs and SHRLEVEL NONE for LOAD RESUME YES jobs.

- SHRLEVEL=(,REFERENCE) indicates that the default is SHRLEVEL NONE for LOAD REPLACE jobs and SHRLEVEL REFERENCE for LOAD RESUME YES jobs.
**Overriding this option**

You can override the value for this option by using the SHRLEVEL command option. For more information about SHRLEVEL options, see “SHRLEVEL” on page 189.

**SMAX=16**

The SMAX option specifies the maximum number of each type of task that can run concurrently. The following table describes the type of tasks that SMAX controls:

<table>
<thead>
<tr>
<th>Phase</th>
<th>Task type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANALYZE</td>
<td>Any</td>
</tr>
<tr>
<td>PRELOAD</td>
<td>Sort, index</td>
</tr>
<tr>
<td>LOAD</td>
<td>Load, index</td>
</tr>
<tr>
<td>COMBINED</td>
<td>Sort, index, probe</td>
</tr>
</tbody>
</table>

The maximum value that you can specify is 16.

When invoking DSNUTILB, LOADPLUS ignores this option.

If the value of the LOADDN option is 7 or more characters and the value of the SMAX option is greater than 9, LOADPLUS reduces the value of the SMAX option to 9.

**Overriding this option**

The MAXSORTS command option overrides the SMAX option only when the MAXSORTS value is smaller than the SMAX value, and only for those tasks that MAXSORTS controls. For more information about the command option, see “MAXSORTS” on page 236.

**SMCORE=(0K,0K)**

The SMCORE option specifies the amount of memory that you want each invocation of BMCSORT to use. The values 0K and 0K tell LOADPLUS to determine the appropriate amount of memory to use for each sort process. For more information, see “Tuning memory use in sort processing” on page 687.
**Note**

BMC strongly recommends that you use the values 0K and 0K for this option. If you use values other than 0K and 0K for this option, LOADPLUS uses these values and ignores the values for the MINSORTMEMORY, MAXSORTMEMORY, and SHORTMEMORY options.

The first value specifies the total amount of memory to use both above and below the 16-megabyte line for each sort process. You can specify this value in either kilobytes or megabytes. The following values are valid for this parameter:

- 0K (or 0M) tells LOADPLUS to determine the appropriate amount.
- Any value from 4096K through 65536K (or 4M through 64M) tells LOADPLUS to use the specified amount.

The second value specifies the amount of memory that is required below the 16-megabyte line for each sort process. You can specify this value in either kilobytes or megabytes. The following values are valid for this parameter:

- 0K (or 0M) tells LOADPLUS to determine the appropriate amount.
- Any value from 256K through 4096K (or 1M through 4M) tells LOADPLUS to use the specified amount.

When invoking DSNUTILB, LOADPLUS ignores this option.

**SORTDEVT=(,SYSALLDA)**

The SORTDEVT option specifies the device type for the sort work files that are allocated dynamically.

The first parameter of this option is the device type to use for non-DSNUTILB load jobs. This parameter overrides the first parameter of the BMCSORT DYNALOC installation option. If the value of the third parameter in the BMCSORT DYNALOC installation option is OFF, specifying a value for this parameter turns BMCSORT dynamic allocation on.

The second parameter of this option is the device type to use for DSNUTILB load jobs. When invoking DSNUTILB, LOADPLUS passes this option to the IBM DB2 LOAD utility for processing.

**Overriding this option**

You can override the values for both parameters of this option by using the SORTDEVT command option.
SORTNUM=32

The SORTNUM option affects the allocation of sort work files in the following cases. You can specify any integer value from 0 through 255.

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS passes this value to the IBM DB2 LOAD utility as the number of sort work files to allocate dynamically. For this type of load, the value must be 2 or greater.

**All other load jobs**

This value is in effect when BMCSORT is allocating your sort work files dynamically.

The following table describes the action that BMCSORT takes for each value that you can specify for this option. The table also provides any additional considerations for these values.

**Table 130: SORTNUM values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Additional considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>BMCSORT honors the value of the third parameter of the BMCSORT DYNALOC installation option. (This parameter tells BMCSORT whether to dynamically allocate sort work files.)</td>
<td>For more information about how this parameter affects dynamic allocation, see “Dynamically allocating SORTWK data sets” on page 447.</td>
</tr>
<tr>
<td>1-32</td>
<td>BMCSORT dynamically allocates the number of sort work files that it needs, up to 32 minus any preallocated sort work files. This number is per sort task.</td>
<td>Preallocated sort work files include sort work files that are allocated in your JCL and any sort work files that LOADPLUS dynamically allocates.</td>
</tr>
<tr>
<td>33-255</td>
<td>BMCSORT dynamically allocates the number of sort work files that it needs, up to the number that you specified minus any preallocated sort work files. This number is per sort task.</td>
<td></td>
</tr>
</tbody>
</table>

If the value of the third parameter in the BMCSORT DYNALOC installation option is OFF, specifying a value greater than 0 for the SORTNUM option turns on BMCSORT dynamic allocation, and BMCSORT allocates sort work files as needed. For information about when BMCSORT allocates your sort work files dynamically, see “SORTWK data sets in LOADPLUS” on page 446.

**Overriding this option**
You can override the value for this option by using the SORTNUM command option.

**SQLDELAY=3**

The SQLDELAY option specifies the number of seconds that LOADPLUS should wait between retry attempts after receiving an SQL -911 return code. This interval is in addition to the time that elapses when DB2 waits for a timeout or deadlock. The number of seconds can range from 1 through 655.

**SQLRETRY=100**

The SQLRETRY option specifies the number of times that LOADPLUS should attempt an SQL statement. The number of attempts can range from 1 through 255.

**STOP@CMT=YES**

_This option applies to SHRLEVEL NONE and SHRLEVEL REFERENCE only._

The STOP@CMT option specifies whether to add the 'AT(COMMIT)' parameter to all DB2 STOP commands that LOADPLUS issues. YES adds the parameter and NO does not.

BMC recommends that you specify the value for STOP@CMT based on your environment and your availability goals. Under certain circumstances, using STOP@CMT=NO might result in the object being placed in STOP pending status (STOPP) and failure of the load. LOADPLUS issues the following message in this case:

```
BMC50266E SPACE 'databaseName.spaceName' CANNOT BE STOPPED. IT MAY BE IN USE
```

This situation might result when you are loading one partition of a partitioned table space while an application thread is using other partitions of the same table space.

By using STOP@CMT=YES, you might avoid this problem, but application threads might be affected. When a thread performs a commit, DB2 might stop the object, although the thread continues to run. For details and implications of the 'AT(COMMIT)' parameter, see the IBM DB2 command reference manual.

**STOPDELAY=1**

The STOPDELAY option specifies the number of seconds that LOADPLUS waits before it checks again to determine whether DB2 has stopped the object. The number of seconds can be any nonzero positive integer.
**STOPRETRY=300**

The STOPRETRY option specifies the number of times that LOADPLUS checks to determine whether DB2 has changed the status of an object from stop pending (STOPP) to stopped (STOP). The number of checks can be any nonzero positive integer.

If you plan to load objects with more than 2000 partitions, BMC recommends that you specify a minimum value of 450.

**TAPEDISP=DELETE**

The TAPEDISP option specifies the final disposition of tape work files when the value of the DELFILES installation option or the DELETEFILES or SYSDISC command option is YES.

If you specify TAPEDISP=DELETE, each tape file will have a disposition of OLD,DELETE,DELETE when the tape is deallocated.

If you specify TAPEDISP=UNCATLG, each tape file will have a disposition of OLD,UNCATLG,UNCATLG when the tape is deallocated. Depending on your tape management environment, using UNCATLG can prevent a tape remount.

**TEMPRALDATA (obsolete)**

In LOADPLUS 11.1 and earlier, this option told LOADPLUS whether to load (by invoking DSNUTILB) a temporal table under certain conditions.

Beginning with version 11.2, LOADPLUS no longer requires or supports the TEMPRALDATA installation option. If your installation options module contains this option, the module compiles successfully but completes with return code 4 and a message indicating that this option is obsolete.

**TOTALPAGEPCT=30**

The TOTALPAGEPCT option controls memory that LOADPLUS allocates to BMCSORT for concurrent sort processing. TOTALPAGEPCT specifies the maximum percentage of total 4-KB pages, as obtained from the system, that LOADPLUS can allocate. You can specify any integer from 0 through 100.

**Note**

LOADPLUS defines total pages as pages that are underutilized and are available for use. Available pages (which you can control with the AVAILPAGEPCT option), are pages that have not been used.
A value of 0 tells LOADPLUS to ignore the number of total pages when allocating sort memory.

A value of 1 through 100 tells LOADPLUS to use up to the specified percentage of total pages when allocating sort memory. For example, TOTALPAGEPCT=50 indicates that LOADPLUS should use no more than 50 percent of the total pages.

**Additional considerations**

The following additional information applies to the TOTALPAGEPCT option:

- Because total pages are subject to more system paging than available pages, changing the value for this option is more likely to affect system performance than changing the value for the AVAILPAGEPCT option.

- When you specify values greater than 0 for both TOTALPAGEPCT and AVAILPAGEPCT, LOADPLUS uses the lesser of the two calculated results as the maximum amount of memory for sort processing.

- If LOADPLUS is unable to perform an optimal sort due to an insufficient number of available or total pages that it is enabled to allocate, the SHORTMEMORY installation option or command option controls the action that LOADPLUS takes.

- When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the TOTALPAGEPCT command option.

**TSSAMPLEPCT=100**

The TSSAMPLEPCT option enables you to specify a percentage of table space pages that you want LOADPLUS to sample when gathering statistics. The following values are valid:

- 1 through 50 tells LOADPLUS to sample the specified percentage of the table space pages.
- 100 tells LOADPLUS to read all table space pages instead of sampling.

**Note**

Values 51 through 99 are not valid.

LOADPLUS ignores the TSSAMPLEPCT option for either of the following types of load jobs:
A DSNUTILB load
An SQLAPPLY load

Overriding this option

You can override the value for this option by using the TSSAMPLEPCT command option.

UNICODE (obsolete)

In LOADPLUS version 11.1 and earlier, this option told LOADPLUS whether to invoke DSNUTILB when the requested translation required it.

Beginning with version 11.2, LOADPLUS no longer requires or supports the UNICODE installation option. If your installation options module contains this option, the module compiles successfully but completes with return code 4 and a message indicating that this option is obsolete.

UPDMAXA=NO

The UPDMAXA option tells LOADPLUS whether to update the MAXASSIGNEDVAL column of SYSIBM.SYSSEQUENCES when loading identity column values from an input file.

**Note**

When generating all identity column values, LOADPLUS automatically updates MAXASSIGNEDVAL, regardless of the value of the UPDMAXA option.

Valid values for this option are YES and NO.

To enable LOADPLUS to update MAXASSIGNEDVAL when UPDMAXA=YES, one of the following authorization IDs must have ALTER privileges on the table that you are loading:

- User ID of the job owner, when UPDMAXA_AUTHID=USER
- INSTALL SYSADM, when UPDMAXA_AUTHID=INSTALLSYSADM

The UPDMAXA_AUTHID option controls which ID must have these privileges.

When invoking DSNUTILB, LOADPLUS ignores this option.

Overriding this option

You can override the value for this option by using the UPDATEMAXA command option.
The UPDMAXA_AUTHID option controls which authorization ID must have ALTER privileges to enable LOADPLUS to update MAXASSIGNEDVAL when UPDATEMAXA YES is in effect. These privileges must be on the table that you are loading.

You can specify the following values:

- **USER** tells LOADPLUS to ensure that the user ID of the job owner has ALTER privileges.
- **INSTALLSYSADM** tells LOADPLUS to ensure that the INSTALL SYSADM authorization ID has ALTER privileges.

**UTILB_COLCCSID (obsolete)**

In LOADPLUS version 11.1, this option told LOADPLUS whether to load (by invoking DSNUTILB) a table that contained columns defined with a CCSID specification.

Beginning with version 11.2, LOADPLUS no longer requires or supports the UTILB_COLCCSID installation option. If your installation options module contains this option, the module compiles successfully but completes with return code 4 and a message indicating that this option is obsolete.

**UTILB_NULLIX (obsolete)**

In LOADPLUS 11.1, this option told LOADPLUS whether to load (by invoking DSNUTILB) table spaces that contained an index that was defined with EXCLUDE NULL KEYS.

Beginning with version 11.2, LOADPLUS no longer requires or supports the UTILB_NULLIX installation option. If your installation options module contains this option, the module compiles successfully but completes with return code 4 and a message indicating that this option is obsolete.

**UXSTATE=SUP**

The UXSTATE option tells LOADPLUS how to invoke DB2 user exits:

- **UXSTATE=SUP** specifies that LOADPLUS should call DB2 user exits (such as EDITPROCs or date/time exits) in supervisor state (and PSW key=7).
- **UXSTATE=PROB** tells LOADPLUS to call DB2 user exits in problem state (and PSW key=7).
The requirements of the exits dictate the UXSTATE setting. Check with the exit author (or vendor) before changing to UXSTATE=PROB.

**WBUFFS=(20,10)**

WBUFFS specifies a multiple of 32 KB to use to define the amount of buffer pool storage for each work (SYSUT1) data set. LOADPLUS allocates as many buffers as will fit into the calculated space.

LOADPLUS uses the first number if you specify only one work data set. LOADPLUS uses the second number for each data set if you specify multiple work data sets. For more information, see “Performance considerations for buffer options” on page 656.

**WORKDDN=SYSUT1**

WORKDDN specifies a default ddname or ddname prefix for the index work data set. When using this name as a prefix, ensure that you allow sufficient bytes for LOADPLUS to add the number of indexes and still have a valid ddname (eight bytes or less). In addition, if you change the default ddname from SYSUT1 and you are not dynamically allocating your SYSUT1 data sets, you must change the name in your JCL. For more information, see “SYSUT1 data sets in LOADPLUS” on page 459.

When invoking DSNUTILB, LOADPLUS uses this option only to pass a value for the &DDNAME variable for the data set name pattern. LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

*Overriding this option*

You can override the value for this option by using the WORKDDN command option.

**WORKUNIT=SYSALLDA**

WORKUNIT specifies the unit to use for a temporary work data set. VIO is an acceptable value for this option.

**XBLKS=3**

XBLKS specifies the number of blocks per receiving process to use for the following types of transfers during the PRELOAD phase or the COMBINED phase:

- Data from the read task to the data process
- Index data from the data process to the index-building process
The minimum value is 3. For more information, see “Performance considerations for the XBLKS installation option” on page 669.

**XBMID=**

XBMID specifies the XBM subsystem (SSID) that LOADPLUS accesses when it uses XBM or SUF. Specify a value for this option if you want to use a specific XBM subsystem for snapshot processing and zIIP processing in LOADPLUS.

The SSID is the unique identifier that was specified when XBM or SUF was installed. If you are using XBM or SUF in a DB2 data sharing environment, you can use the value of the XBMGROUP parameter instead of the XBM SSID. The XBMGROUP name is the name of the cross-system coupling facility (XCF) group that is defined to the XBM subsystem, and its default value is XBMGROUP.

If you specify an XBM subsystem, it must be:

- Available
- At a supported maintenance level
- Enabled for the required function

If you do not specify an XBM subsystem (either here or with the XBMID installation option), LOADPLUS automatically searches for an XBM subsystem that meets the same criteria. If you have multiple subsystems that meet these criteria, LOADPLUS can use any one of these subsystems.

For more information about using XBM or SUF with LOADPLUS, see “XBM and SUF considerations in LOADPLUS” on page 160.

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the XBMID command option.

**XML (obsolete)**

In LOADPLUS 11.1 and earlier, this option told LOADPLUS whether to load XML data when DSNUTILB was required.

Beginning with version 11.2, LOADPLUS no longer requires or supports the XML installation option. If your installation options module contains this option, the module compiles successfully but completes with return code 4 and a message indicating that this option is obsolete.
**XMLAVGSIZE=10M**

The XMLAVGSIZE option provides LOADPLUS with an estimate of the actual size of your XML data. LOADPLUS uses this information to balance the number of tasks needed to load the base table, load the XML table space, and build the indexes. LOADPLUS also uses this information when allocating XML copy data sets.

Specify a number for the estimated size of your XML data, optionally followed by a unit of measure (K, M, or G). The default unit of measure is K. The minimum value that you can specify is 1K. The maximum value that you can specify is 2G, 2048M, or 2097152K.

When invoking DSNUTILB, LOADPLUS ignores this option.

*Overriding this option*

You can override the value for this option either at the global level, by using the XMLAVGSIZE option, or at the field level, by using the AVGSIZE field option on the CLOBF or XML field specification. The field-level option overrides the global option.

**ZEROROWRC=0**

ZEROROWRC allows you to specify a default user-defined return code that LOADPLUS uses upon successful completion of the job when LOADPLUS does not load any rows. You can specify any integer from 0 through 7.

LOADPLUS always ends with the highest return code assigned during the load job. For more information, see “Return code hierarchy in LOADPLUS” on page 158.

When invoking DSNUTILB, LOADPLUS ignores this option.

*Overriding this option*

You can override the value for this option by using the IFZEROROWS command option.

**ZIIP=ENABLED**

The ZIIP option tells LOADPLUS whether to attempt to use IBM z Integrated Information Processors (zIIPs). LOADPLUS 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 LOADPLUS jobs.

You can specify one of the following values:
ENABLED tells LOADPLUS 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 LOADPLUS to not attempt to use zIIP processing.

To enable and use zIIP processing with LOADPLUS, 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 or command option. For more information, see “XBMID” on page 269.

For more information about using XBM or SUF with LOADPLUS, see “XBM and SUF considerations in LOADPLUS” on page 160.

When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the ZIIP command option.

---

**Dynamic allocation installation options for LOADPLUS**

The $AMUDYNA macros produce the installation options for dynamic allocation (one macro for each data set type).

The following table shows the options contained in the $AMUDYNA macros. For each option, the table provides the value that ships with this version of LOADPLUS (or No value if the option is shipped without a value), a brief description, and a reference to more details. If an option ships with no value, the table shows a recommended value or example value.

Your $730DOP2 member must contain exactly 18 $AMUDYNA macros. Each macro must specify a different DDTYPE.
Table 131: LOADPLUS installation options for dynamic allocation

<table>
<thead>
<tr>
<th>Option</th>
<th>Shipped value</th>
<th>Brief description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDTYPE</td>
<td>Each macro has a different default value. For example, the default value for local primary copy data sets is LOCPFCPY.</td>
<td>Data set type to which the remaining dynamic allocation options apply</td>
<td>“DDTYPE=LOAD” on page 763</td>
</tr>
<tr>
<td>ACTIVE</td>
<td>Each DDTYPE has a different default value. For example, the default value for DDTYPE LOAD is YES.</td>
<td>Whether to activate dynamic allocation</td>
<td>“ACTIVE=YES” on page 763</td>
</tr>
<tr>
<td>AVGVOLSP</td>
<td>((30000,TRK), (30000,TRK))</td>
<td>Average space available for data on volumes that are used for dynamic allocation</td>
<td>“AVGVOLOSP=((30000,TRK),(30000,TRK))” on page 765</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>(NONE,NONE)</td>
<td>SMS data class names</td>
<td>“DATACLAS=(NONE,NONE)” on page 766</td>
</tr>
<tr>
<td>DSNPAT</td>
<td>Each DDTYPE has a different default value. For example, the default value for DDTYPE LOAD is &amp;&amp;UID.&amp;JOBNAME.&amp;TS.&amp;DDNAME.</td>
<td>Default pattern for creating data set names</td>
<td>“DSNPAT=&amp;&amp;UID.&amp;JOBNAME.&amp;TS.&amp;DDNAME” on page 766</td>
</tr>
<tr>
<td>DSNTYPE</td>
<td>(NONE,NONE)</td>
<td>Data set type</td>
<td>“DSNTYPE=(NONE,NONE)” on page 771</td>
</tr>
<tr>
<td>EXPDT</td>
<td>No value</td>
<td>Example value: 2006365</td>
<td>“EXPDT=” on page 772</td>
</tr>
<tr>
<td>GDGEMPTY</td>
<td>NO</td>
<td>Whether to uncatalog all data sets when the limit is reached</td>
<td>“GDGEMPTY=NO” on page 773</td>
</tr>
<tr>
<td>GDGLIMIT</td>
<td>5</td>
<td>Maximum number of generation data sets in a group</td>
<td>“GDGLIMIT=5” on page 773</td>
</tr>
<tr>
<td>GDGSCRATCH</td>
<td>NO</td>
<td>Whether to keep DSCB information when the data sets are uncataloged</td>
<td>“GDGSCRATCH=NO” on page 774</td>
</tr>
<tr>
<td>IFALLOC</td>
<td>USE</td>
<td>Whether to use or free data sets that are specified in the JCL</td>
<td>“IFALLOC=USE” on page 774</td>
</tr>
<tr>
<td>Option</td>
<td>Shipped value</td>
<td>Brief description</td>
<td>Reference</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>MAXEXTSZ</td>
<td>((0,K),(0,K))</td>
<td>Maximum size for any extents that LOADPLUS allocates</td>
<td>“MAXEXTSZ=((0,K),(0,K))” on page 775</td>
</tr>
<tr>
<td>MGMTCLAS</td>
<td>(NONE,NONE)</td>
<td>SMS management class names</td>
<td>“MGMTCLAS=(NONE,NONE)” on page 776</td>
</tr>
<tr>
<td>RETPD</td>
<td>No value</td>
<td>Retention period for certain data sets</td>
<td>“RETPD=” on page 776</td>
</tr>
<tr>
<td>SIZEPCT</td>
<td>(100,100)</td>
<td>Percent of the calculated space to allocate</td>
<td>“SIZEPCT=(100,100)” on page 777</td>
</tr>
<tr>
<td>SMS</td>
<td>NO</td>
<td>Whether to pass SMS classes to SMS</td>
<td>“SMS=NO” on page 778</td>
</tr>
<tr>
<td>SMSUNIT</td>
<td>NO</td>
<td>Whether to pass the value for UNIT to SMS</td>
<td>“SMSUNIT=NO” on page 778</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>(NONE,NONE)</td>
<td>SMS storage class names</td>
<td>“STORCLAS=(NONE,NONE)” on page 779</td>
</tr>
<tr>
<td>THRESHLD</td>
<td>0</td>
<td>Whether to use secondary units, classes, number and size of volumes, and extent size</td>
<td>“THRESHLD=0” on page 779</td>
</tr>
<tr>
<td>UNIT</td>
<td>(SYSALLDA,SYSALLDA)</td>
<td>Default device names to use during dynamic allocation</td>
<td>“UNIT=(SYSALLDA,SYSALLDA)” on page 780</td>
</tr>
<tr>
<td>UNITCNT</td>
<td>(0,0)</td>
<td>Number of devices to dynamically allocate</td>
<td>“UNITCNT=(0,0)” on page 781</td>
</tr>
<tr>
<td>VOLCNT</td>
<td>(25,25)</td>
<td>Largest number of volumes to process</td>
<td>“VOLCNT=(25,25)” on page 782</td>
</tr>
</tbody>
</table>

**Descriptions of dynamic allocation installation options for LOADPLUS**

This topic describes each of the dynamic allocation installation options.

Most of these options can also be specified by using the DDTYPE command option, allowing you to override the defaults. For details, see “Dynamic allocation options” on page 393.
**DDTYPE=LOAD**

DDTYPE specifies the data set type for which you are establishing dynamic allocation options. The following table lists the valid values for this option. Each instance of the $AMUDYNA macro must contain a different value for this option.

### Table 132: DDTYPE option values

<table>
<thead>
<tr>
<th>DDTYPE value</th>
<th>Corresponding default ddname</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD</td>
<td>SORTOUT</td>
</tr>
<tr>
<td>WORK</td>
<td>SYSUT1</td>
</tr>
<tr>
<td>SORTWORK</td>
<td>SORTWK</td>
</tr>
<tr>
<td>DISCARD</td>
<td>SYSDISC</td>
</tr>
<tr>
<td>ERROR</td>
<td>SYSERR</td>
</tr>
<tr>
<td>SYSMAP</td>
<td>SYSMAP</td>
</tr>
<tr>
<td>LOCPFCPY</td>
<td>BMCCPY</td>
</tr>
<tr>
<td>LOCBFCPY</td>
<td>BMCCPZ</td>
</tr>
<tr>
<td>REMPFCPY</td>
<td>BMRCY</td>
</tr>
<tr>
<td>REMBFCPY</td>
<td>BMRCZ</td>
</tr>
<tr>
<td>LOCPLCPY</td>
<td>LLPnnnnn</td>
</tr>
<tr>
<td>LOCBLCPY</td>
<td>LLBnnnnn</td>
</tr>
<tr>
<td>REMPLCPY</td>
<td>LRPnnnnn</td>
</tr>
<tr>
<td>REMBLCPY</td>
<td>LRBnnnnn</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>XLPnnnnn</td>
</tr>
<tr>
<td>LOCBXCPY</td>
<td>XLBnnnnn</td>
</tr>
<tr>
<td>REMPXCPY</td>
<td>XRPnnnnn</td>
</tr>
<tr>
<td>REMBXCPY</td>
<td>XRBnnnnn</td>
</tr>
</tbody>
</table>

**Overriding this option**

You can override each instance of the DDTYPE installation option by using the DDTYPE command option.

**ACTIVE=YES**

ACTIVE tells LOADPLUS whether to dynamically allocate the specified data sets. YES activates dynamic allocation and NO inactivates dynamic allocation. Each
DDTYPE has its own default value for the ACTIVE option, as shown in the following table:

Table 133: ACTIVE option default values by DDTYPE

<table>
<thead>
<tr>
<th>DDTYPE</th>
<th>ACTIVE value</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD</td>
<td>YES</td>
</tr>
<tr>
<td>WORK</td>
<td>YES</td>
</tr>
<tr>
<td>SORTWORK</td>
<td>NO</td>
</tr>
<tr>
<td>DISCARD</td>
<td>YES</td>
</tr>
<tr>
<td>ERROR</td>
<td>YES</td>
</tr>
<tr>
<td>SYSMAP</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPFCPY</td>
<td>YES</td>
</tr>
<tr>
<td>LOCBFCPY</td>
<td>NO</td>
</tr>
<tr>
<td>REMPFCPY</td>
<td>NO</td>
</tr>
<tr>
<td>REMBFCPY</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPBCPY</td>
<td>YES</td>
</tr>
<tr>
<td>LOCBLCPY</td>
<td>NO</td>
</tr>
<tr>
<td>REMPLCPY</td>
<td>NO</td>
</tr>
<tr>
<td>REMBLCPY</td>
<td>NO</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>YES</td>
</tr>
<tr>
<td>LOCBXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>REMPXCPY</td>
<td>NO</td>
</tr>
<tr>
<td>REMBXCPY</td>
<td>NO</td>
</tr>
</tbody>
</table>

**Additional considerations**

The following considerations apply to the ACTIVE option:

- For a DSNUTILB load, ACTIVE must be YES for all work file DDTYPEs. If you also specify COPY YES in your load job, ACTIVE must be YES for at least the local primary copy data set (DDTYPE LOCPFCPY). LOADPLUS ignores any copy or work file data set allocation in your JCL.

**Note**

All copy data sets for a DSNUTILB load are dynamically allocated, even if you specify ACTIVE YES for only the local primary copy data set.
If you specify COPY YES when loading LOB or XML data, you must enable dynamic allocation for at least the primary local copy data set (DDTYPE LOCLCPY or DDTYPE LOCPXCPY). In general, LOADPLUS ignores any LOB or XML copy data set allocation in your JCL. However, if you specify any DD statements in your JCL that use one of the default LOB or XML copy prefixes (Table 99 on page 437), LOADPLUS terminates.

When LOADPLUS runs in a CHANGE MANAGER worklist environment, it ignores the ACTIVE option in your installation options module. LOADPLUS dynamically allocates your data sets only if CHANGE MANAGER supplies the ACTIVE YES syntax.

**Overriding this option**

You can override the value for this option by using the ACTIVE command option.

**AVGVOLSP=((30000,TRK),(30000,TRK))**

AVGVOLSP enables you to specify the average amount of space that is available on each device. Use this option to reflect the average space available on volumes that are eligible to contain the dynamically allocated data set.

Specify one or two integer values, and include the unit of measure as follows:

- K for kilobytes
- TRK for tracks (the default)
- CYL for cylinders

LOADPLUS uses AVGVOLSP only when you specify AUTO for the corresponding first or second parameter of the VOLCNT installation or command option.

LOADPLUS uses the second parameter when the value for the THRESHLD installation or command option is exceeded. This value applies to the secondary device type that you specify with the UNIT option (or, if applicable, through SMS).

**Restrictions**

The following restrictions apply to the AVGVOLSP option:

- The AVGVOLSP option is not valid for DDTYPE SORTWORK.
- When invoking DSNUTILB, LOADPLUS ignores this option.

**Additional considerations**

Note the following additional information about the AVGVOLSP option:
You should not use AVGVOLSP to specify the maximum space on all devices or volumes unless the volumes to be used are empty.

If you specify a value for AVGVOLSP that is too small, LOADPLUS computes a value for VOLCNT that is too large. If you specify a value for AVGVOLSP that is too large, LOADPLUS computes a value for VOLCNT that is too small.

**Overriding this option**

You can override the value for this option by using the AVGVOLSP command option.

**DATAclas=(NONE,NONE)**

DATAclas specifies the primary and secondary SMS data classes that LOADPLUS uses for an SMS allocation. The two class names must be valid SMS data class names, not exceeding eight characters each, or NONE.

**DSNuTILB load jobs**

When invoking DSNUTILB, LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

If building a secondary TEMPLATE control statement for your copy data sets, LOADPLUS includes the secondary data class in that template. For information about when LOADPLUS builds a secondary template, see “THRESHLD=0” on page 779.

**Overriding this option**

You can override the value for this option by using the DATAclas command option.

**DSNPAT=&&UID.&&JOBNAME.&&TS.&&DDNAME**

DSNPAT specifies the pattern that LOADPLUS uses to generate data set names during dynamic data set allocation.

For sort work data sets only, you can also specify the value NONE. Do not enclose NONE in single quotes with the DSNPAT installation option (even though the quotes are required in the DSNPAT command option).

Each DDTYPE has its own default value for the DSNPAT option, as shown in the following table:
### Table 134: DSNPAT option default values by DDTYPE

<table>
<thead>
<tr>
<th>DDTYPE</th>
<th>DSNPAT value</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD</td>
<td>&amp;&amp;UID.&amp;&amp;JOBNAME.&amp;&amp;TS.&amp;&amp;DDNAME</td>
</tr>
<tr>
<td>WORK</td>
<td>&amp;&amp;UID.&amp;&amp;JOBNAME.&amp;&amp;TS.&amp;&amp;DDNAME</td>
</tr>
<tr>
<td>SORTWORK</td>
<td>NONE</td>
</tr>
<tr>
<td>DISCARD</td>
<td>&amp;&amp;UID.&amp;&amp;JOBNAME.&amp;&amp;TS.&amp;&amp;DDNAME</td>
</tr>
<tr>
<td>ERROR</td>
<td>&amp;&amp;UID.&amp;&amp;JOBNAME.&amp;&amp;TS.&amp;&amp;DDNAME</td>
</tr>
<tr>
<td>SYSMAP</td>
<td>&amp;&amp;UID.&amp;&amp;JOBNAME.&amp;&amp;TS.&amp;&amp;DDNAME</td>
</tr>
<tr>
<td>LOCPFCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..F&amp;&amp;PART..T&amp;&amp;TIME</td>
</tr>
<tr>
<td>LOCBFCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..F&amp;&amp;PART..T&amp;&amp;TIME</td>
</tr>
<tr>
<td>REMPFCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..F&amp;&amp;PART..T&amp;&amp;TIME</td>
</tr>
<tr>
<td>REMBFCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..F&amp;&amp;PART..T&amp;&amp;TIME</td>
</tr>
<tr>
<td>LOCPLCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..T&amp;&amp;TIME</td>
</tr>
<tr>
<td>LOCBLCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..T&amp;&amp;TIME</td>
</tr>
<tr>
<td>REMPLCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..T&amp;&amp;TIME</td>
</tr>
<tr>
<td>REMBLCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..T&amp;&amp;TIME</td>
</tr>
<tr>
<td>LOCPXCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..F&amp;&amp;PART..T&amp;&amp;TIME</td>
</tr>
<tr>
<td>LOCBXCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..F&amp;&amp;PART..T&amp;&amp;TIME</td>
</tr>
<tr>
<td>REMPXCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..F&amp;&amp;PART..T&amp;&amp;TIME</td>
</tr>
<tr>
<td>REMBXCPY</td>
<td>&amp;&amp;UID.&amp;&amp;DDNAME.&amp;&amp;TS..F&amp;&amp;PART..T&amp;&amp;TIME</td>
</tr>
</tbody>
</table>

### General considerations

The following considerations apply to the DSNPAT option:

- The pattern that you specify in your DSNPAT option must allow LOADPLUS to generate unique data set names. If LOADPLUS encounters non-unique data set names, processing terminates. Note the following considerations:

  - With multiple SORTOUT and SYSUT1 files, you must include the &DDNAME variable to generate unique names.

  - For copy data sets, you might need to include additional variables, such as &VCAT, &DATEJ, or &TIME4 to generate unique names across multiple loads.

  - If you plan to run concurrent LOADPLUS jobs or run multiple LOADPLUS steps in a worklist environment, BMC recommends that you include additional...
variables in the default pattern to ensure unique names. A value such as the following ensures that LOADPLUS generates unique names:

\[
\text{DSNPAT} = \&\text{UID} \ldots \&\text{UTILPFX} \ldots \&\text{DDNAME} \ldots \&\text{DATE} \ldots \&\text{TIME}
\]

- You can use text or any of the symbolic variables from Table 135 on page 768 to construct your pattern. You can also provide user-defined variables from a user exit routine (specified with the DSNUEXIT installation or command option). For more information about using symbolic variables to construct a data set name pattern, see “DSNPAT” on page 414.

- When specifying a pattern in your installation options, you must precede each variable with an additional ampersand (&) in your pattern.

- The maximum total length allowed for a data set name is 44 bytes.

- LOADPLUS removes any trailing blanks in the resolved pattern.

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS includes this pattern in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. Any variables that you include in your pattern for this type of load must be either valid for the TEMPLATE control statement or translatable (as shown in Table 135 on page 768) to a valid TEMPLATE variable. User-defined variables are not valid for a DSNUTILB load. For more information about TEMPLATE control statements, see the documentation for the DB2 LOAD utility.

**Overriding this option**

You can also specify the data set name pattern by using the DSNPAT command option, which overrides any default. Additional ampersands are not allowed with the command option. However, if you use the keyword NONE with the DSNPAT command option, you must enclose NONE with single quotation marks.

**Table 135: Symbolic variables for DSNPAT installation option**

<table>
<thead>
<tr>
<th>Symbolic variable</th>
<th>Definition</th>
<th>Length of result</th>
<th>DSNUTILB load</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;DATE</td>
<td>Current date (in the format MMDDYY)</td>
<td>6 bytes</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;DATEJ</td>
<td>Current Julian date (in the format YYYYDDD)</td>
<td>7 bytes</td>
<td>Variable translated to the IBM &amp;JDATE variable</td>
</tr>
<tr>
<td>&amp;DB</td>
<td>Database containing the space being used for this data set allocation</td>
<td>8 bytes maximum</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;DDNAME</td>
<td>Ddname being used for this data set allocation</td>
<td>8 bytes maximum</td>
<td>Value passed</td>
</tr>
<tr>
<td>Symbolic variable</td>
<td>Definition</td>
<td>Length of result</td>
<td>DSNUTILB load</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>&amp;GRPNM</td>
<td>DB2 data sharing group name, or, in a non-data-sharing environment, the DB2 SSID</td>
<td>4 bytes</td>
<td>Value passed</td>
</tr>
<tr>
<td>&amp;JDATE</td>
<td>Current Julian date (in the format &quot;YYDDD&quot;)</td>
<td>5 bytes</td>
<td>Variable translated to the IBM &amp;JDATE(3,5) variable</td>
</tr>
<tr>
<td>&amp;JOBNAME</td>
<td>JOB name used in the JCL</td>
<td>8 bytes maximum</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;PART</td>
<td>Partition being used for this copy data set allocation. You can use this variable for any data set. However, LOADPLUS substitutes the partition number only for copy data sets. For all other data sets, LOADPLUS substitutes the value 000.</td>
<td>3 bytes for table spaces with 999 partitions or less, 4 bytes for table spaces with 1000 through 4096 partitions</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;REPLACE</td>
<td>LOAD REPLACE being done (Y or N)</td>
<td>1 byte</td>
<td>Job terminated</td>
</tr>
<tr>
<td>&amp;RESUME</td>
<td>LOAD RESUME being done (Y or N)</td>
<td>1 byte</td>
<td>Job terminated</td>
</tr>
<tr>
<td>&amp;SSID</td>
<td>DB2 subsystem ID</td>
<td>4 bytes</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;STEPNAME</td>
<td>STEP name used in the JCL. LOADPLUS ignores PROC names.</td>
<td>8 bytes maximum</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;TIME</td>
<td>Current time (in the format &quot;HHMMSS&quot;)</td>
<td>6 bytes</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;TIME4</td>
<td>Current time (in the format &quot;HHMM&quot;)</td>
<td>4 bytes</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;TS</td>
<td>Table space containing the table specified in your LOAD command</td>
<td>8 bytes maximum</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;USERID or &amp;UID</td>
<td>Job user ID. You must have a security package to use this variable.</td>
<td>8 bytes maximum</td>
<td>Variable passed</td>
</tr>
<tr>
<td>&amp;UTIL a</td>
<td>BMC utility ID. LOADPLUS truncates utility IDs longer than 8 bytes</td>
<td>8 bytes maximum</td>
<td>Variable translated to the IBM &amp;UTILID variable</td>
</tr>
<tr>
<td>&amp;UTILPFX a</td>
<td>BMC utility ID prefix</td>
<td>8 bytes maximum</td>
<td>Value passed</td>
</tr>
<tr>
<td>&amp;UTILSFX a</td>
<td>BMC utility ID suffix</td>
<td>8 bytes maximum</td>
<td>Value passed</td>
</tr>
</tbody>
</table>
Symbolic variable | Definition | Length of result | DSNUTILB load
--- | --- | --- | ---
&VCAT | VCATNAME specified in the DB2 catalog for the table space that you are loading; or, if the table space is partitioned, the VCAT name from the first partition that you are loading | 8 bytes | Job terminated

Utility IDs that include special characters might cause LOADPLUS to generate invalid data set names. For more information, see page 406.

For the following dynamically allocated data sets, you can also specify a pattern that contains a GDG name:

- Copy data sets
- SYSDISC files
- SYSERR files
- SYSMAP files

Each DDTYPE must have a different GDG base.

The GDG format that you use to construct a data set name is the same as the format that you use in your JCL when you allocate data sets with DD statements. Simply append the generation number in parentheses. The open parenthesis tells LOADPLUS that the pattern is a GDG name. The generation number must be an integer from 1 through 255.

If the base does not exist, LOADPLUS creates it for you using everything in the pattern up to the open parenthesis as the base name. For more information about GDG names and options, see “Generating data set names in LOADPLUS” on page 80.

The following example shows a valid GDG name:

```
'&UTILPFX.&DDNAME..COPY(+1)'
```

If you are using a substitution variable as the last variable before the open parenthesis, you must include a period before the open parenthesis. The following example shows this usage:

```
'&UTILPFX.&DDNAME.(+1)'
```

For copy data sets, each partition must have a different GDG base if you specify COPYLVL PART on the LOAD command. To specify a pattern that includes a
partition, the partition must not be in parentheses. The following example shows a valid name:

`'&UTILPFX.&DDNAME..P&PART.(+1)'`

You cannot specify a pattern that contains a PDS name. The following example shows an invalid name:

`'&UTILPFX.&DDNAME..(P&PART)'`

**Overriding this option**

You can override the value for this option by using the DSNPAT command option. For more information and guidelines for specifying data set name patterns, see “DSNPAT” on page 414.

**DSNTYPE=(NONE,NONE)**

The DSNTYPE option enables you to specify the type of data set that you want LOADPLUS to allocate. If you specify a second value, LOADPLUS uses that value when the value that is in effect for the THRESHLD option is exceeded.

The following values are valid:

- **NONE** (the default) tells LOADPLUS to not use any extended attributes for this data set allocation.

- **LARGE** tells LOADPLUS to allocate this data set as a large format sequential data set. This option enables data sets larger than 65,535 tracks.

- **BASIC** tells LOADPLUS to allocate this data set as a basic sequential data set. This data set will be limited to 65,535 tracks.

- **EXTREQ** tells LOADPLUS to allocate this data set as an extended format data set. This option is ignored if either of the following conditions exists:
  - The SMS option is NO.
  - The DDTYPE is SORTWORK.

- **EXTPREF** indicates that you prefer that LOADPLUS allocate this data set as an extended format data set. If this allocation is not possible, the data set is allocated as a basic format data set. This option is ignored if either of the following conditions exists:
  - The SMS option is NO.
  - The DDTYPE is SORTWORK.

**Restrictions**
The following restrictions apply to the DSNTYPE option:

- The DSNTYPE option is not valid for tape data sets.
- When invoking DSNUTILB, LOADPLUS ignores this option.

**Overriding this option**

You can override the value for this option by using the DSNTYPE command option.

**EXPDT=**

The EXPDT option specifies the expiration date for the following data sets that you are dynamically allocating:

- Copy data sets
- SYSDISC data sets
- SYSERR data sets
- SYSMAP data sets

LOADPLUS ignores this option for any other data sets.

The date must be either blank or in the format `yyyyddd`:

- `yyyy` is the 4-digit year (1900 through 2155).
- `ddd` is the 3-digit Julian day (000 through 366).

When you specify the EXPDT installation option, it takes precedence over the RETPD installation option.

**Additional considerations**

The following considerations apply to the EXPDT option:

- If DELETEFILES YES is in effect, LOADPLUS deletes your SYSERR data set only if the expiration date is earlier than the current date.

- If DELETEFILES YES SYSDISC YES is in effect and you have no discarded records, LOADPLUS deletes your SYSDISC data set only if the expiration date is earlier than the current date.

- When invoking DSNUTILB, LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. For more information about TEMPLATE control statements, see the documentation for the DB2 LOAD utility.

**Overriding this option**
You can override the value for this option with either the EXPDT or RETPD command option.

**GDGEMPTY=NO**

GDGEMPTY=NO indicates that the system uncatalogs only the oldest GDG data set when the GDGLIMIT is reached.

If you specify GDGEMPTY=YES, the system uncatalogs all preexisting generations of this data set when the limit is reached.

This option applies only to the following types of data sets that you are dynamically allocating as a GDG data set:

- Copy data sets
- SYSDISC data sets
- SYSERR data sets
- SYSMAP data sets

LOADPLUS honors this option only when creating the GDG base. LOADPLUS ignores this option for any data sets other than the ones listed.

When invoking DSNUTILB, LOADPLUS ignores this option.

**GDGLIMIT=5**

Specify GDGLIMIT to indicate the number of generations to keep for the following types of data sets that you are dynamically allocating as a GDG data set:

- Copy data sets
- SYSDISC data sets
- SYSERR data sets
- SYSMAP data sets

You can specify an integer from 1 through 255.

LOADPLUS honors this option only when creating the GDG base. LOADPLUS ignores this option for any data sets other than the ones listed.

When invoking DSNUTILB, LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

**Overriding this option**

You can override the value for this option by using the GDGLIMIT command option.
**GDGSCRATCH=NO**

GDGSCRATCH=NO specifies that the system should not delete an entry that is uncataloged as a result of the GDGEMPTY option.

If you specify GDGSCRATCH=YES, the system deletes the GDG entry from the volume’s table of contents (VTOC) when uncataloging the data set. The space on the volume is then available to other users.

This option applies only to the following types of data sets that you are dynamically allocating as a GDG data set:

- Copy data sets
- SYSDISC data sets
- SYSERR data sets
- SYSMAP data sets

LOADPLUS honors this option only when creating the GDG base. LOADPLUS ignores this option for any data sets other than the ones listed.

When invoking DSNUTILB, LOADPLUS ignores this option.

**IFALLOC=USE**

IFALLOC tells LOADPLUS how to handle any data sets that are specified in your JCL:

- **USE** tells LOADPLUS to use the data sets that you allocated in your JCL. If the number of data sets that you specify in your JCL is insufficient for processing, LOADPLUS (or BMCSORT in the case of sort work files) dynamically allocates the additional data sets that your job needs.

- **FREE** tells LOADPLUS to free the data sets that you allocated in your JCL and use only dynamically allocated data sets.

LOADPLUS ignores this option in the following cases:

- DSNUTILB load jobs
- LOB and XML copy data sets

*Overriding this option*

You can override the value for this option by using the IFALLOC command option.
MAXEXTSZ=((0,K),(0,K))

For any extent that LOADPLUS allocates for a dynamically allocated data set, this option allows you to specify the maximum allowable value for the primary space allocation. Because the secondary quantity cannot exceed the primary quantity, MAXEXTSZ also controls the maximum secondary quantity.

Specify one of the following values for the first value of each parameter:

- 0 if you do not want to set a limit for space allocation
- An integer for the units specified with the UNIT installation or command option

For the second value of each parameter, specify a unit of measure as follows:

- K for kilobytes (the default)
- TRK for tracks
- CYL for cylinders

LOADPLUS uses the second parameter when the value specified for the THRESHLD installation or command option is exceeded. This value applies to the secondary device type that you specify with the UNIT option (or, if applicable, through SMS).

Regardless of the amount of space that LOADPLUS determines that it needs, LOADPLUS will not allocate more than your specified MAXEXTSZ limit for either the primary or the secondary quantity:

- If the amount of required space that LOADPLUS calculates is greater than the MAXEXTSZ limit for the primary quantity, LOADPLUS uses the secondary extents to hold the remainder of the required primary space.

- If the amount of required space that LOADPLUS calculates cannot be accommodated because of MAXEXTSZ restrictions, the job might terminate with an out-of-space condition on the data set.

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS includes the first parameter of this option as the MAXPRIME value in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. For DSNUTILB, the unit of measure is always cylinders.

If building a secondary TEMPLATE control statement for your copy data sets, LOADPLUS includes the second parameter of this option as the MAXPRIME value in that template. For information about when LOADPLUS builds a secondary template, see “THRESHLD=0” on page 779.

**Additional considerations**
Note the following additional information about the MAXEXTSZ option:

- The MAXEXTSZ option is not valid for DDTYPE SORTWORK.
- LOADPLUS ignores MAXEXTSZ when you specify SMS YES.
- LOADPLUS checks the value of MAXEXTSZ after applying SIZEPCT to the allocation amount.

**Overriding this option**

You can override the value for this option by using the MAXEXTSZ command option.

**MGMTCLAS= (NONE, NONE)**

MGMTCLAS specifies the primary and secondary SMS management classes that LOADPLUS uses for an SMS allocation. The two class names must be valid SMS management class names, not exceeding eight characters each, or NONE.

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. For more information about TEMPLATE control statements, see the documentation for the DB2 LOAD utility.

If building a secondary TEMPLATE control statement for your copy data sets, LOADPLUS includes the secondary management class in that template. For information about when LOADPLUS builds a secondary template, see “THRESHLD=0” on page 779.

**Overriding this option**

You can override the value for this option by using the MGMTCLAS command option.

**RETPD=**

Use RETPD to specify the retention period (in days) for the following data sets that you are dynamically allocating:

- Copy data sets
- SYSDISC data sets
- SYSERR data sets
- SYSMAP data sets
LOADPLUS ignores this option for any other data sets.

The number of days must be blank or in the range 0 through 9999. A blank value means that LOADPLUS does not use a retention period.

Additional considerations

The following considerations apply to the RETPD option:

■ If DELETEFILES YES is in effect, LOADPLUS deletes your SYSERR data set only if RETPD is 0 or blank.

■ If DELETEFILES YES SYSDISC YES is in effect and you have no discarded records, LOADPLUS deletes your SYSDISC data set only if RETPD is 0 or blank.

■ When invoking DSNUTILB, LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. For more information about TEMPLATE control statements, see the documentation for the DB2 LOAD utility.

Overriding this option

You can override the value for this option by using the RETPD or EXPDT command option. If you specify the EXPDT installation option, the EXPDT installation option takes precedence over the RETPD installation option.

SIZEPCT=(100,100)

SIZEPCT tells LOADPLUS what percentages of the primary and secondary space to use for allocation. The values that you specify must be greater than 0.

The first number indicates the percentage of the primary quantity (calculated by LOADPLUS) to allocate. The second number indicates the percentage of the secondary quantity to allocate.

DSNUTILB load jobs

When invoking DSNUTILB, LOADPLUS includes the first parameter of this option as the PCTPRIME value in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. If you specify a value greater than 100, LOADPLUS converts that value to 100.

Overriding this option

You can override the value for this option by using the SIZEPCT command option.
**SMS=NO**

SMS tells LOADPLUS whether to pass the SMS classes and extended data set types (EXTREQ and EXTPREF) to SMS during dynamic allocation. Whether LOADPLUS actually performs an SMS allocation depends on your site. You can specify one of the following values:

- **YES** tells LOADPLUS to pass SMS classes and extended data set types during dynamic allocation. The following considerations apply to SMS=YES:
  
  — When you specify SMS=YES, LOADPLUS ignores the values that you specify for the MAXEXTSZ option.
  
  — To pass the UNIT value to SMS during dynamic allocation, also specify SMSUNIT=YES.
  
  — When invoking DSNUTILB, LOADPLUS includes the SMS classes in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

- **NO** tells LOADPLUS not to pass SMS classes and extended data set types during dynamic allocation.

When you specify SMS=NO, LOADPLUS always passes the UNIT value during dynamic allocation.

*Overriding this option*

You can override the value for this option by using the SMS command option.

**SMSUNIT=NO**

SMSUNIT tells LOADPLUS whether to pass the UNIT value in the SMS allocation parameter list during dynamic allocation. LOADPLUS does not modify any other parameters based on this option. If the value of the SMS option is NO, LOADPLUS ignores the SMSUNIT option.

You can specify the following values:

- **NO** tells LOADPLUS not to pass the value for the UNIT option.
- **YES** tells LOADPLUS to pass the value for the UNIT option.

When invoking DSNUTILB, LOADPLUS includes the UNIT option values in the TEMPLATE control statements that it builds for the IBM DB2 LOAD utility.

*Overriding this option*

You can override the value for this option by using the SMSUNIT command option.
STORCLAS=(NONE,NONE)

STORCLAS specifies the primary and secondary SMS storage classes that LOADPLUS uses for an SMS allocation. The two class names must be valid SMS storage class names, not exceeding eight characters each, or NONE.

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS includes this information in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility. For more information about TEMPLATE control statements, see the documentation for the DB2 LOAD utility.

If building a secondary TEMPLATE control statement for your copy data sets, LOADPLUS includes the secondary storage class in that template. For information about when LOADPLUS builds a secondary template, see “THRESHLD=0” on page 779.

**Overriding this option**

You can override the value for this option by using the STORCLAS command option.

**THRESHLD=0**

This option allows you to specify a threshold value, in kilobytes, above which LOADPLUS applies secondary values to allocated data sets.

LOADPLUS tests this threshold for each data set to be allocated. If the size for a data set is greater than the threshold, LOADPLUS performs the following tasks:

- When SMS is NO, LOADPLUS uses the secondary values of the following options for the allocated data sets:
  - UNIT (unit name)
  - UNITCNT (unit count)
  - VOLCNT (volume count)
  - AVGVOLSP (average volume space)
  - MAXEXTSZ (extent size)
  - DSNTYPE (data set type), unless DSNTYPE is EXTREQ or EXTPREF

- When SMS is YES, LOADPLUS uses the secondary values of the following options (or passes them to SMS if applicable):
  - STORCLAS, MGMTCLAS, or DATACLAS (class name for the SMS classes)
  - VOLCNT (volume count)
  - AVGVOLSP (average volume space)
— DSNTYPE (data set type)
— If SMSUNIT is YES
— UNIT (unit name)
— UNITCNT (unit count)

If you specify 0 or if the threshold is not exceeded, LOADPLUS uses the primary values for these options.

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS handles this option differently, depending on the type of data set that you are allocating:

- For copy data sets, LOADPLUS translates this option to a LIMIT value in the TEMPLATE control statement that LOADPLUS builds for the IBM DB2 LOAD utility. LOADPLUS also builds the secondary TEMPLATE control statement to which the DB2 LOAD utility will switch when the LIMIT is exceeded.
  LOADPLUS assumes that the value that you supply for THRESHLD is in kilobytes. LOADPLUS translates the value to the appropriate value and unit of measure for the LIMIT keyword. Note the following additional information about this value:
  - LOADPLUS rounds down to the nearest whole value.
  - If you specify a value that would cause LOADPLUS to translate to a value less than 1 cylinder, LOADPLUS builds the template with a LIMIT value of 1 CYL.

- For all other data sets, LOADPLUS ignores this option.

**Additional considerations**

If you use THRESHLD to send larger data sets to tape, consider setting the MAXTAPE option to limit the number of tapes that you use.

For examples of using the THRESHLD option with other dynamic allocation options, see “THRESHLD” on page 411.

**Overriding this option**

You can override the value for this option by using the THRESHLD command option.

**UNIT=(SYSALLDA,SYSALLDA)**

For non-SMS-managed data sets, the UNIT option specifies the primary and secondary unit names that LOADPLUS uses for dynamic allocation. These unit names may not exceed eight characters each.
For SMS-managed data sets when you specify SMS YES and SMSUNIT YES, this option supplies the unit names that LOADPLUS passes in the SMS allocation parameter list. DSNUTILB load jobs function differently, as described in the DSNUTILB load jobs section.

The THRESHLD command or installation option controls which of the specified unit names LOADPLUS selects.

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS includes the first parameter in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

If building a secondary TEMPLATE control statement for your copy data sets, LOADPLUS includes the second UNIT parameter in that template. For information about when LOADPLUS builds a secondary template, see “THRESHLD=0” on page 779.

**Overriding this option**

You can override the value for this option by using the UNIT command option.

**UNITCNT=(0,0)**

This option allows you to specify the number of devices to allocate when dynamically allocating data sets. Valid values are 0 through 59. A value of 0 tells LOADPLUS to use the system default.

LOADPLUS uses the second value when the value for THRESHLD is exceeded. This value applies to the secondary device type that you specify with the UNIT option (or, if applicable, through SMS).

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS includes the first parameter in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

If building a secondary TEMPLATE control statement for your copy data sets, LOADPLUS includes the second UNITCNT parameter in that template. For information about when LOADPLUS builds a secondary template, see “THRESHLD=0” on page 779.

**Additional considerations**

Note the following additional information about the UNITCNT option:

- This option is not available for DDTYPE SORTWORK.
To avoid performance problems, specify only the number of devices that you need.

*Overriding this option*

You can override the value for this option by using the UNITCNT command option.

**VOLCNT=(25,25)**

This option enables you to specify the maximum number of volumes to use during dynamic data set allocation. The following values are valid for this option:

- 0 tells LOADPLUS to not specify a volume count for dynamic allocation.
- Integer values 1 through 255 specify the number of volumes.
- AUTO tells LOADPLUS to compute the volume count based on the amount of data, adjusted for the estimated space required (based on, for example, the values of the SIZEPCT and ENUMROWS options).

LOADPLUS computes the volume count by dividing the size estimate by the value specified for the AVGVOLSP installation or command option.

If you specify a second value, LOADPLUS uses that value when the value for the THRESHLD installation or command option is exceeded. This value applies to the secondary device type that you specify with the UNIT option (or, if applicable, through SMS).

**DSNUTILB load jobs**

When invoking DSNUTILB, LOADPLUS includes the first parameter in the TEMPLATE control statement that it builds for the IBM DB2 LOAD utility.

If building a secondary TEMPLATE control statement for your copy data sets, LOADPLUS includes the second VOLCNT parameter in that template. For information about when LOADPLUS builds a secondary template, see “THRESHLD=0” on page 779.

**Additional considerations**

Note the following additional information about the VOLCNT option:

- This option is not available for DDTYPE SORTWORK. If you specify a value for this option for sort work files, LOADPLUS changes the value to 1.
- To avoid performance problems, specify only the number of volumes that you need.
For an SMS-managed data set, BMC recommends that you specify 0 if your ACS routines are set up to provide a volume count.

If you plan to load LOB or XML data, BMC recommends increasing the value of this option from the shipped value.

If you anticipate loading a large number of partitions, consider changing the value of this option to AUTO to avoid data set allocation limitations.

**Overriding this option**

You can override the value for this option by using the VOLCNT command option.

---

**DYNALOC installation option**

The DYNALOC installation option of BMCSORT provides information for dynamically allocating SORTWK data sets.

BMCSORT deallocates these data sets at the end of each sort. The content of the $AUPSMAC macro in $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
```

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 136 on page 784 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 136: 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>
<tr>
<td>Position 3</td>
<td>This parameter tells the BMC product whether to dynamically allocate SORTWK files. BMC recommends that you not change this value.</td>
<td>ON</td>
<td>■ ON dynamically allocates SORTWK.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ OFF does not dynamically allocate SORTWK.</td>
</tr>
<tr>
<td>Position 4</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>ON</td>
<td>Do not change this value.</td>
</tr>
<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>
<tr>
<td>Position 7</td>
<td>This parameter specifies the DASD type with the smallest track capacity that a dynamically allocated SORTWK data set might encounter at your site.</td>
<td>3390</td>
<td>■ 3380, track capacity of 47968</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ 3390, track capacity of 56664</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ 9345, track capacity of 46456</td>
</tr>
<tr>
<td>Parameter name or position</td>
<td>Description</td>
<td>Initial value</td>
<td>Valid values</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>SC</td>
<td>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.</td>
<td>blank</td>
<td>Use any valid DFSMS storage class.</td>
</tr>
</tbody>
</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 137 on page 787 lists the tables that each utility uses and each table’s default name and synonym.

<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>
</tbody>
</table>
## Overview of 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>BMCLGRNX</td>
<td>CMN_BMCLGRNX</td>
<td>BMC_BMCLGRNX</td>
<td>- COPY PLUS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Log Master</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- RECOVER PLUS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- RECOVERY MANAGER</td>
</tr>
<tr>
<td>BMCSYNC</td>
<td>CMN_BMCSYNC</td>
<td>BMC_BMCSYNC</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>- DASD MANAGER PLUS (BMCSTATS)</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>- RECOVERY MANAGER</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>BMCTRANS</td>
<td>CMN_BMCTRANS</td>
<td>BMC_BMCTRANS</td>
<td>- Log Master</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- 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
  - BMXCOPY

- 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 787):

  ```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'
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 138 on page 793 describes the contents of the BMCDICT table.
Table 138: 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&lt;br/&gt;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:

```sql
DELETE
FROM creatorName.CMN_BMC_DICT
WHERE UTILID NOT IN
  (SELECT UTILID FROM creatorName.CMN_BMC_UTIL);
```

*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 139 on page 794 describes the 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>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) or the PART option. Note the following conditions:</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 DSNUTILIB)

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 < 'yyyymm-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 140 on page 797 describes the contents of the BMCLGRNX table.

Table 140: 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 (mmdddy)</td>
</tr>
</tbody>
</table>
BMCSYNC table

The BMCSYNC table contains information about the status of the objects that the currently executing utilities are accessing.

Table 141 on page 798 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 141: 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 BUILDDINDEX command reads and deletes the row.</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.
<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
</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>
</tbody>
</table>
| UTILNAME    | CHAR(8)   | Name of the executing utility:  
|             |           | ■ CHECK  
|             |           | ■ COPY  
|             |           | ■ STATS  
|             |           | ■ LOAD  
|             |           | ■ RECOVER  
|             |           | ■ REORG  
|             |           | ■ UNLOAD  |
| SHRLEVEL    | CHAR(1)   | Degree to which utilities can share this object:  
|             |           | ■ Blank means that no status is requested, and any other utility can obtain any status.  
|             |           | ■ S allows sharing among any number of SHRLEVEL S utilities.  
|             |           | ■ X indicates that exclusive control is required. No other utility can run with SHRLEVEL X.  
|             |           | For more information, see “Shared access levels of BMC utilities” on page 66. |
| STATUS      | CHAR(1)   | Status of the utility or object:  
|             |           | ■ Blank (indicates no processing has been done)  
|             |           | ■ C (for CHECK PLUS, indicates checked)  
|             |           | ■ L (for LOADPLUS, indicates loaded)  
|             |           | ■ U (for UNLOAD PLUS, indicates unloaded)  
|             |           | ■ R (for REORG PLUS, indicates reloaded)  
<p>|             |           | DASD MANAGER PLUS does not use this column. |</p>
<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCOUNT</td>
<td>INTEGER</td>
<td>Number of rows or keys processed in the current phase. DASD MANAGER PLUS does not use this column.</td>
</tr>
<tr>
<td>DDNAME</td>
<td>CHAR(8)</td>
<td>Check, load, unload, or work ddname. DASD MANAGER PLUS does not use this column.</td>
</tr>
<tr>
<td>BLOCKS</td>
<td>INTEGER</td>
<td>Number of blocks for the check, load, unload, or work data set. DASD MANAGER PLUS does not use this column.</td>
</tr>
<tr>
<td>ORIG_STATUS</td>
<td>CHAR(8)</td>
<td>Encoded representation of the original DB2 status of the space. <em>(RECOVER PLUS)</em> This column restores the DB2 status of a space after recovery, if necessary. DASD MANAGER PLUS does not use this column.</td>
</tr>
<tr>
<td>EXTRBA</td>
<td>CHAR(10)</td>
<td><em>(RECOVER PLUS)</em> Log point at which this space was externalized. RECOVER PLUS serialization logic uses this column. The other utilities do not use this column. <strong>Note:</strong> RECOVER PLUS no longer uses EXTRBA.</td>
</tr>
<tr>
<td>STATE</td>
<td>LONG VARCHAR</td>
<td>Restart information for the space. For example, the STATE indicates the object state and sync information. DASD MANAGER PLUS does not use this column.</td>
</tr>
<tr>
<td>INSTANCE</td>
<td>SMALLINT</td>
<td><em>(RECOVERY MANAGER and RECOVER PLUS)</em> Instance number of the current base objects (table and index). The default value is 1. The other utilities do 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 808.

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 798). 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 142: 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 143 on page 806 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 143: 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 DEFAULT</td>
<td>Transaction end time</td>
</tr>
<tr>
<td>ACTION</td>
<td>SMALLINT</td>
<td>Code indicating what recovery, if any, has been performed on the transaction</td>
</tr>
</tbody>
</table>

**BMCUTIL table**

The BMCUTIL table contains information about utilities that are currently running or started.

Table 144 on page 809 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 144: 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>
</tbody>
</table>
| STATUS      | CHAR(1)   | Execution status of the utility:  
|             |           | ■ A (active, not executing command)  
|             |           | ■ I (initializing)  
|             |           | ■ P (pausing or pause-stopped)  
|             |           | ■ S (stopped)  
|             |           | ■ T (terminating)  
|             |           | ■ X (executing command)  
|             |           | *(DASD MANAGER PLUS)* The value for this column is always X. |
| UTILNAME    | CHAR(8)   | Name of the executing utility:  
|             |           | ■ CHECK  
|             |           | ■ COPY  
|             |           | ■ STATS  
|             |           | ■ LOAD  
|             |           | ■ RECOVER  
|             |           | ■ REORG  
|             |           | ■ UNLOAD  
| PHASE       | CHAR(8)   | Current phase of the utility  
|             |           | COPY PLUS does not use this column.  
| USERID      | CHAR(8)   | User ID executing the utility  
<p>| SSID        | CHAR(4)   | DB2 subsystem where the utility is running  |</p>
<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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:

```
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 145 on page 812 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 145: 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>
<tr>
<td>Column name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>ICTYPE</td>
<td>CHAR(1)</td>
<td>Operation type:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ F (COPY FULL YES; for COPY PLUS, online consistent copies)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ I (COPY FULL NO)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ W (REORG LOG NO)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ B (REBUILD INDEX)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ P (POINT-IN-TIME RECOVERY)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ C (for COPY PLUS version 7.3 and earlier, online consistent copies)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ s (used by COPY PLUS to track system pages)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ m (indicates that the table space was exported by the COPY PLUS EXPORT command or migrated by the RECOVER PLUS IMPORT command)</td>
</tr>
<tr>
<td>ICDATE</td>
<td>CHAR(6)</td>
<td>Date of the entry (yymmdd)</td>
</tr>
<tr>
<td>START_RBA</td>
<td>VARCHAR(10)</td>
<td>The relative byte location of a point in the DB2 recovery log</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The indicated point as follows:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ For ICTYPE F, the starting point for all updates since the image copy was taken</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ For COPY_TYPE O, the minimum of the consistent point and the oldest inflight URID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ (RECOVERY MANAGER) For ICTYPE C, the consistent log point for the copy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— RBA for non-data-sharing systems</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— LRSN for data sharing systems</td>
</tr>
<tr>
<td>FILESEQNO</td>
<td>INTEGER</td>
<td>Tape file sequence number of the copy</td>
</tr>
<tr>
<td>DEVTYPE</td>
<td>CHAR(8)</td>
<td>Type of device on which the copy resides</td>
</tr>
<tr>
<td>Column name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| IBMREQD     | CHAR(1)   | Whether the row came from the basic machine-readable material (MRM) tape:  
  ■ N (NO)  
  ■ Y (YES) |
| DSNAME      | CHAR(44)  | Name of the data set  
  If STYPE V, DSNAME 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)  
  ■ Blank (data set contains local system main data or is not one of multiple copies) |
<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’00000000000000000000’ (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) 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>
<tr>
<td>Column name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NOTE_VALUE</td>
<td>CHAR(4)</td>
<td>Encoded value that quickly locates data for a specific space in a cabinet copy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td></td>
<td>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;
```
RULES installation option

This chapter describes the RULES installation option, including how handling differs with the different values, and how to avoid error conditions.

Overview of the RULES installation option

This appendix describes the RULES installation option of the LOADPLUS for DB2 product.

The RULES installation option is set during installation, and you cannot override it at runtime. You can specify either RULES=STANDARD or RULES=BMC.

Note

If you specify FORMAT UNLOAD, FORMAT BMC, or FORMAT BMCUNLOAD, LOADPLUS changes the value of this option to BMC, regardless of the value that you specify.
If you specify FORMAT CSV, LOADPLUS changes the value of this option to STANDARD, regardless of the value that you specify.

The information in this appendix uses a single sample table to illustrate the RULES option. The following statement creates this sample table:

```
CREATE TABLE SAMPLE_TABLE
    (CHAR_COL    CHAR(1)      NOT NULL WITH DEFAULT,
     DEC_COL     DECIMAL(5,0),
     DATE_COL    DATE)
IN DATABASE.TABLESPACE
```

Comparison operators allowed

RULES=STANDARD has more restrictions on comparison operators than RULES=BMC.
Each option permits different comparison operators for use in WHEN, NULLIF, and DEFAULTIF predicates, as shown in the following table:

Table 146: Comparison operators for RULES option

<table>
<thead>
<tr>
<th>Operators</th>
<th>Description</th>
<th>RULES=STANDARD</th>
<th>RULES=BMC</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td>Less than</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or equal</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>=</td>
<td>Equal</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>Not equal</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>¬ =</td>
<td>Not equal</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>&gt; =</td>
<td>Greater than or equal</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>IN</td>
<td>Equal to any</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NOT IN</td>
<td>Not equal to all</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

WHEN, NULLIF, and DEFAULTIF constants handling

The RULES option affects how LOADPLUS handles constants in your WHEN, DEFAULTIF, or NULLIF clauses.

These differences are as follows:

- With RULES=BMC, LOADPLUS expects the constant to be in EBCDIC format and translates the constant to the CCSID of the table space. If the constant is specified as a hexadecimal string, the string must be in the internal format of the output.

- With RULES=STANDARD, LOADPLUS expects the constant to be in EBCDIC format and translates the constant to the CCSID of the input. If the constant is specified as a hexadecimal string, the string must be in the internal format of the input.
WHEN, NULLIF, and DEFAULTIF processing order

The RULES option affects the order in which LOADPLUS performs WHEN, NULLIF, and DEFAULTIF processing.

These differences are as follows:

- With RULES=BMC, LOADPLUS first completes, on the input data, any data conversion and conversion due to NULLIF and DEFAULTIF conditions. LOADPLUS then compares values to the WHEN condition based on the type of comparison:
  - For predicates with column names on the left side, LOADPLUS compares the WHEN condition to the converted data.
  - For predicates with (start:end) or field names on the left side, LOADPLUS compares the WHEN condition to the original input data.

- With RULES=STANDARD, LOADPLUS compares the values in the input file to the WHEN condition. LOADPLUS then completes, on the resulting records, any data conversion and conversion due to NULLIF and DEFAULTIF conditions.

The discussion in this section uses the following example:

```
LOAD DATA REPLACE INTO TABLE SAMPLE_TABLE
  WHEN CHAR_COL = 'X'
  (CHAR_COL POS(1:1) DEFAULTIF CHAR_COL='X'
   DEC_COL POS(2:4) DEC(5,0),
   DATE_COL POS(5:8) DATE NULLIF DATE_COL=' ')
```

With RULES=BMC in effect, LOADPLUS performs the NULLIF or DEFAULTIF processing first to determine whether to set CHAR_COL to a blank because it is defined as NOT NULL WITH DEFAULT. Next, LOADPLUS performs the WHEN comparison to determine whether to load the record into the table. For any record with an 'X' in column 1, LOADPLUS changes the 'X' to a blank and discards the record.

With RULES=STANDARD in effect, the opposite process occurs. LOADPLUS first performs the WHEN comparison to determine whether to load the record into the table, and, if so, continues with the NULLIF or DEFAULTIF data conversion. For any record with an 'X' in column 1, LOADPLUS changes the 'X' to a blank and loads the record into the table.
Error conditions with RULES=STANDARD

The topics in this section describe some of the errors that you might encounter under RULES=STANDARD and how to avoid them.

Invalid operator

Under RULES=STANDARD, you might encounter an invalid operator error.

The following statement creates an error condition when RULES=STANDARD is the installation option, because you cannot use the less than operator (<):

```
LOAD DATA REPLACE INTO TABLE SAMPLE_TABLE
  (CHAR_COL POS(1:1) CHAR(1)
   DEC_COL POS(2:4) DEC(5,0)NULLIF DEC_COL<'5',
   DATE_COL POS(5:8) DATE)
```

LOADPLUS issues the following error:

```
BMC51415E FOR 'NULLIF/DEFAULTIF' FIELD 'DEC_COL', ONLY '=','¬ =','<>','IN', AND 'NOT IN' COMPARISONS ARE ALLOWED.
```

Because RULES=BMC allows use of the less-than operator, you can avoid this error by changing your RULES specification.

Invalid constant in predicate

RULES=STANDARD requires that all constants used in WHEN, NULLIF, or DEFAULTIF predicates be character strings or hexadecimal strings enclosed in quotes.

The following statement produces an error because LOADPLUS converts the character string '1' to a numeric value for comparison with the decimal column, resulting in a constant that does not meet the requirement for RULES=STANDARD:

```
LOAD DATA REPLACE INTO TABLE SAMPLE_TABLE
  (CHAR_COL POS(1:1) CHAR(1)
   DEC_COL POS(2:4) DEC(5,0)NULLIF DEC_COL='1',
   DATE_COL POS(5:8) DATE)
```

LOADPLUS issues the following error:

```
BMC51416E FOR 'NULLIF/DEFAULTIF' FIELD 'DEC_COL', CONSTANT MUST BE A CHARACTER STRING OR A HEX STRING
```
Because RULES=BMC does not have this requirement, you can avoid this error by changing your RULES specification.

**Error conditions with RULES=BMC**

The topics in this section describes some of the errors that you might encounter under RULES=BMC and how to avoid them.

**Invalid constant in WHEN statement**

Under RULES=BMC, you might encounter an invalid constant error in your WHEN statement.

In the following example, LOADPLUS attempts to compare a character string constant (' ') with a numeric column:

```sql
LOAD DATA REPLACE INTO SAMPLE_TABLE
    WHEN DEC_COL= ' '
```

LOADPLUS issues the following error message:

```
BMC50122E FOR ‘WHEN’ COLUMN ‘DEC_COL’, CONSTANT IS WRONG TYPE OR COLUMN IS NOT NULLABLE: ‘ ’
```

**Invalid constant in NULLIF statement**

Under RULES=BMC, you might encounter an invalid constant error in your NULLIF statement.

In the following example, LOADPLUS attempts to perform NULLIF processing by comparing a character string constant (' ') with a date column:

```sql
LOAD DATA REPLACE INTO TABLE SAMPLE_TABLE
    (CHAR_COL POS(1:1) CHAR(1),
     DEC_COL POS(2:4) DEC(5,0),
     DATE_COL POS(5:8) DATE NULLIF DATE_COL=' ')
```

LOADPLUS issues the following error message:

```
BMC50123E FOR ‘NULLIF/DEFAULTIF’ COLUMN ‘DATE_COL’, CONSTANT IS NOT IN RANGE OF COLUMN DATA TYPE: ‘ ’
```
Preventing these errors

You can use one of the following methods to avoid the errors when RULES=BMC:

- Use the RULES=STANDARD installation option.

- Use starting and ending (start:end) column positions in your LOAD command as shown in the following example:

```sql
LOAD DATA REPLACE INTO TABLE SAMPLE_TABLE
(CHAR_COL POS(1:1) CHAR(1),
 DEC_COL POS(2:4) DEC(5,0),
 DATE_COL POS(5:8) DATE NULLIF (5:8)=' ')
```
LOADPLUS user exits

This chapter describes user-written exit routines that you can use with LOADPLUS.

Overview of LOADPLUS user exits

The LOADPLUS for DB2 product allows you to further customize your load processing through user-written exit routines.

These user exits provide LOADPLUS with user-defined variables that you can use to construct data set name patterns with the DSNPAT installation option or command option. See “DSNPAT” on page 414 for details about the DSNPAT option.

LOADPLUS supports user exits written in the following languages:

- Assembler
- COBOL II
- IBM Language Environment COBOL (LE COBOL)
- C
- IBM Language Environment C (LE C)

You supply the module name and program language in either the DSNUEXIT installation or command option. The DSNUEXIT command option overrides any value in the installation options.

Depending on the type of load that you are running, LOADPLUS invokes your user-written exit routine at the beginning of either the PRELOAD phase or the COMBINED phase.

Requirements and restrictions for LOADPLUS user exits

LOADPLUS user exits have the following requirements:
The library in which this exit resides must be included in your system’s LINKLIST or in your JOBLIB or STEPLIB.

For COBOL II and C programs, the appropriate runtime libraries must be authorized and in your LINKLIST, JOBLIB, or STEPLIB.

For LE COBOL and LE C programs, the appropriate language environment runtime libraries must be authorized and in your LINKLIST, JOBLIB, or STEPLIB.

Your routine must be reentrant.

You must have dynamic data set allocation active to direct LOADPLUS to invoke your exit routine.

You cannot use these exit routines with a DSNUTILB load.

Using sample LOADPLUS user exits

Source code for the sample LOADPLUS user exits is distributed in the LOADPLUS HLQ.LLQSAMP library (where HLQ is the high-level qualifier set during installation and LLQ is the low-level qualifier or prefix set during installation).

The following table lists the sample exit names and the command option to use to invoke the sample exit:

<table>
<thead>
<tr>
<th>User exit name</th>
<th>Description</th>
<th>LOAD command option to invoke</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMUEDSNA</td>
<td>Assembler sample exit</td>
<td>DSNUEXIT AMUEDSNA ASM</td>
<td>“LOADPLUS Assembler user exit” on page 828</td>
</tr>
<tr>
<td>AMUEDSN2</td>
<td>LE COBOL and COBOL II sample exit</td>
<td>DSNUEXIT AMUEDSN2 LE_COBOL or DSNUEXIT AMUEDSN2 COBOL2</td>
<td>“LOADPLUS COBOL II and LE COBOL user exit” on page 839</td>
</tr>
<tr>
<td>AMUEDSC</td>
<td>C (standard) sample exit</td>
<td>DSNUEXIT AMUEDSC C</td>
<td>“LOADPLUS C user exit” on page 847</td>
</tr>
<tr>
<td>AMUEDSL</td>
<td>LE C sample exit</td>
<td>DSNUEXIT AMUEDSL LE_C</td>
<td>“LOADPLUS LE C user exit” on page 854</td>
</tr>
</tbody>
</table>
To use the sample user exits

1. Copy the appropriate member as the base code for your exit routine.

2. Edit the exit routine as needed.

3. Compile, or assemble and link your resulting exit into an authorized library so that it can be loaded during execution.

Return codes for LOADPLUS user exits

LOADPLUS communicates with your exit by passing parameters to the exit and receiving a return code and user-defined variables from the exit.

The following list describes the location of the return code for each type of exit:

- For an assembler exit, the return code is the contents of register 15.
- For a COBOL exit, the return code is set in the RETURN-CODE variable.
- For a C exit, the return code is the value that is returned by the return function.

Valid return codes and their results are:

- 0 indicates a good return
  Processing continues.

- other codes indicate an error
  The utility terminates.

Rules for the user-defined variables created in LOADPLUS user exits

The LOADPLUS user exits return user-defined variables that you can use with the DSNPAT installation or command option to construct data set name patterns.

LOADPLUS uses these patterns and the values that are returned for any user-defined variables to create data set names during dynamic data set allocation.

The following rules apply when using a user-defined exit routine to create your variables:
Variable names must comply with the following rules:

— Must begin with an underscore character
— Must begin in the leftmost byte of the XPUVNAME field
— Must be padded on the right with blanks if the variable name is shorter than 9 bytes
— Must not contain embedded blanks
— Nonblank characters after the underscore must consist of the following alphanumeric or national characters:
  — A through Z,
  — 0 through 9,
  — #, @, $

Variable data must comply with the following rules:

— Must begin in the leftmost byte of the XPUVDATA field
— Must be padded on the right with blanks if the variable data is shorter than 8 bytes
— Must not contain embedded blanks
— Nonblank characters must consist of the following characters:
  — A through Z,
  — 0 through 9,
  — #, @, $, -, .

LOADPLUS Assembler user exit

The topics in this section provide a sample exit parameter block, variable mapping block, and exit that you can use to write a user exit in assembler.
Exit parameter block DSECT

The exit parameter block DSECT contains both input and output fields.

Input fields pass vital information to the user exit, such as the database name and user ID. If you make any modifications to these input fields, LOADPLUS disregards them on return. The output fields pass information about your user variables back to LOADPLUS.

Figure 31: Assembler exit parameter block

```
*---------------------------------------------------------------------*
*  AMUDSNXP DEFINES THE USER VARIABLE USER EXIT PARM BLOCK.           *
*  NOTE:                                                               *
*   YOU MAY NOT MODIFY THE FIELDS IN FRONT OF THE USER AREA.          *
*   YOU MAY NOT MODIFY THE 'STRUCTURE' OF THIS DSECT.                 *
*---------------------------------------------------------------------*
AMUDSNXP DSECT ,          PARMS PASSED TO EXIT
  *  INPUT AREA
  * XPJOBN   DS    CL8   JOBNAME
  * XPSTEP   DS    CL8   STEPNAME
  * XPDBNAME DS    CL8   DATABASE NAME
  * XPSPNAME DS    CL8   SPACE NAME
  * XPRESUME DS    CL1   RESUME (Y/N)
  * XPREPLAC DS    CL1   REPLACE (Y/N)
  * DS    CL2   RESERVED FOR LOADPLUS
  * XPUSER   DS    CL8   USER ID
  * XPSSID   DS    CL4   DB2 SUBSYSTEM ID
  * XPDATE   DS    CL6   UTILITY EXECUTION DATE MMDDYY
  * XPTIME   DS    CL6   UTILITY EXECUTION TIME HHMMSS
  * XPUTILID DS    CL16  UTILITY ID
  * XPDATE8  DS    CL8   UTILITY EXECUTION DATE MMDDYYYY
  * XPGRPNM  DS    CL4   DATA SHARING GROUP NAME
  * XPCAT    DS    CL8   VCAT NAME (FROM 1ST PART IF PARTITND)
  * XPDATEJ  DS    CL7   UTILITY EXECUTION DATE CCYYDDD
  * DS    CL13  RESERVED FOR LOADPLUS
  *
  * USER WORK AREA
  *
  * XPUSRW0   DS    F   USER WORD 1
  * XPUSRW2   DS    F   USER WORD 2
  * XPUSRW3   DS    F   USER WORD 3
  * XPUSRW4   DS    F   USER WORD 4
  * XPUSRW5   DS    F   USER WORD 5
  * XPUSRW6   DS    F   USER WORD 6
  * XPUSRW7   DS    F   USER WORD 7
  * XPUSRW8   DS    F   USER WORD 8
  *
  * OUTPUT AREA
  *
  * XPUVAREA DS  100CL17  USER VARIABLE AREA
  * XPUVENT# EQU   100  NUMBER OF ENTRIES IN OUTPUT AREA
  * XP$    EQU   *-AMUDSNXP
```

The following table describes the major DSECT fields:
## Table 148: Assembler user exit parameter block DSECT fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XPJOBN</td>
<td>Contains the job name, up to 8 bytes</td>
</tr>
<tr>
<td>XPSTEP</td>
<td>Contains the step name, up to 8 bytes</td>
</tr>
<tr>
<td>XPDBNAME</td>
<td>Contains the database name, up to 8 bytes</td>
</tr>
<tr>
<td>XPSPNAME</td>
<td>Contains the name of the table space or index space from the LOAD command, up to 8 bytes</td>
</tr>
<tr>
<td>XPRESUME</td>
<td>Contains an indication of whether a LOAD RESUME is being performed: Y (yes) or N (no)</td>
</tr>
<tr>
<td>XPREPLAC</td>
<td>Contains an indication of whether a LOAD REPLACE is being performed: Y (yes) or N (no)</td>
</tr>
<tr>
<td>XPUSER</td>
<td>Contains the user ID of the user running the LOADPLUS utility, up to 8 bytes</td>
</tr>
<tr>
<td>XPSSID</td>
<td>Contains the DB2 subsystem ID, 4 bytes</td>
</tr>
<tr>
<td>XPDATETIME</td>
<td>Contains the date of the execution of the utility, in the format MMDDYY, 6 bytes</td>
</tr>
<tr>
<td>XPTIME</td>
<td>Contains the time of the execution of the utility, in the format HHMMSS, 6 bytes</td>
</tr>
<tr>
<td>XPUTILID</td>
<td>Contains the utility ID, up to 16 bytes</td>
</tr>
<tr>
<td>XPDATE8</td>
<td>Contains the date of the execution of the utility, in the format MMDDYYYY, 8 bytes</td>
</tr>
<tr>
<td>XPGRPNM</td>
<td>Contains the DB2 data sharing group name</td>
</tr>
<tr>
<td></td>
<td>In a non-data-sharing environment, the field contains the DB2 subsystem ID.</td>
</tr>
<tr>
<td>XPVCAT</td>
<td>Contains the VCATNAME specified in the DB2 catalog for the table space being loaded, or for the first partition if the table space is partitioned</td>
</tr>
<tr>
<td>XPDATETIMEJ</td>
<td>Contains the Julian date of the execution of the utility, in the format CCYYDDD, 7 bytes</td>
</tr>
<tr>
<td>XPUSRWD1...XPUSRWD8</td>
<td>Provides work space, 8 parameters, up to 4 bytes each</td>
</tr>
<tr>
<td>XPUVAREA</td>
<td>Defines the area that contains user-defined variable information</td>
</tr>
<tr>
<td></td>
<td>See “Rules for the user-defined variables created in LOADPLUS user exits” on page 827 for details about establishing user-defined variables.</td>
</tr>
<tr>
<td>XPUVENT#</td>
<td>Equates to the maximum number of entries in the output variable area</td>
</tr>
<tr>
<td></td>
<td>There is a limit of 100 entries.</td>
</tr>
</tbody>
</table>
Variable mapping block DSECT

The variable mapping block DSECT defines the output area of the exit parameter DSECT.

Using this DSECT allows you to easily address the variable definition table. The number of entries in this table (each entry consists of both a variable name and its value) must not exceed 100.

**Figure 32: Assembler variable mapping block**

```assembly
XPUVARS DSECT, MAP A VARIABLE ENTRY
XPUVNAME DS CL9 VARIABLE NAME
XPUVDATA DS CL8 VARIABLE DATA
XPUVENT$ EQU *-XPUVNAME LENGTH OF EACH ENTRY
```

The following table describes the major DSECT fields:

**Table 149: Assembler user exit variable mapping block DSECT fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XPUVNAME</td>
<td>The name of the variable that you are defining</td>
</tr>
<tr>
<td>XPUVDATA</td>
<td>The value of the variable that you are defining</td>
</tr>
<tr>
<td>XPUVENT$</td>
<td>The length of each of these variable entries in the table</td>
</tr>
</tbody>
</table>
Sample Assembler user exit

This topic illustrates a sample user exit written in Assembler.

Figure 33: Sample Assembler user exit

```
AMUEDSNA TITLE 'AMUEDSNA - LOADPLUS USER EXIT USER EXAMPLE - V81'
AMUEDSNA CSECT
AMUEDSNA AMODE 31
AMUEDSNA RMODE 24

*---------------------------------------------------------------------*
*                        D I S C L A I M E R                          *
*---------------------------------------------------------------------*
*                                                                     *
*  THIS IS A SAMPLE LOADPLUS USER EXIT.  THIS EXIT WOULD BE USED      *
*  IN ORDER TO DEFINE USER VARIABLES FOR BUILDING DATASET NAMES FOR   *
*  DYNAMIC WORKFILE ALLOCATION WITH THE DSNPAT KEYWORD.               *
*                                                                     *
*  THIS EXIT WILL ONLY BE INVOKED WHEN IT IS SPECIFICALLY NAMED       *
*  IN THE AMU$OPTS DSNEXIT PARAMETER OR IN THE DSNEXIT PARAMETER OF    *
*  THE LOAD COMMAND.  THE SPECIFICATION IN THE LOAD COMMAND WILL      *
*  OVERRIDE THE SPECIFICATION IN THE AMU$OPTS MACRO.  IF THE NAME      *
*  SPECIFIED IS 'NONE' WITHOUT THE SINGLE QUOTE MARKS, THE EXIT WILL   *
*  NOT BE INVOKED.                                                    *
*                                                                     *
*  NOTE: PLEASE REVIEW THE DOCUMENTATION IN THE REFERENCE MANUAL,     *
*  AND THE FOLLOWING USAGE NOTES PRIOR TO IMPLEMENTING THIS EXIT.     *
*                                                                     *
*  PLEASE CALL BMC SOFTWARE TECHNICAL SUPPORT WITH ANY QUESTIONS YOU  *
*  MAY HAVE IN THIS AREA.                                             *
*                                                                     *
*  PHONE: 1-800-537-1813                                              *
*                                                                     *
*---------------------------------------------------------------------*

*                                                                     *
*  AMUEDSNA IS CALLED ONCE AT THE BEGINNING OF EITHER THE PRELOAD      *
*  PHASE OR THE COMBINED PHASE.                                       *
*                                                                     *
*  WHEN INVOKED, R1  CONTAINS THE ADDRESS OF A USER EXIT BLOCK        *
*  DESCRIBED BY THE AMUDSNXP DSECT.                                   *
*  R13 CONTAINS THE ADDRESS OF THE CALLER'S SAVE AREA                 *
*  R14 CONTAINS THE CALLER'S RETURN ADDRESS                            *
*  R15 CONTAINS THE ENTRY POINT ADDRESS OF THIS EXIT                  *
*  UPON EXIT  R15 CONTAINS THE RETURN CODE FROM THIS EXIT             *
*            R15=0     GOOD RETURN                                     *
*            R15=OTHER ERROR RETURN, TERMINATE UTILITY                *
*                                                                     *
*  AMUDSNXP DSECT  THE AREA MAPPED BY THIS DSECT CONTAINS BOTH INPUT   *
*  AND OUTPUT FIELDS.  INPUT FIELDS ARE PROVIDED TO PASS VITAL        *
*  INFORMATION TO THIS EXIT.  ANY MODIFICATIONS TO THESE FIELDS WILL  *
*  BE DISREGARDED UPON RETURN TO THE CALLER.  THE OUTPUT FIELDS ARE   *
*  PROVIDED TO ALLOW THE EXIT TO PASS INFORMATION BACK TO THE CLASSER.*
*  *                                                          *
```

832  LOADPLUS for DB2 Reference Manual
* EJECT
* * INTERNAL REGISTER USAGE
* *
* 0 -
* 1 - ON ENTRY TO MODULE = A(AMUDSNXP BLOCK) COPIED TO R10
* 2 -
* 3 -
* 4 -
* 5 -
* 6 -
* 7 -
* 8 -
* 9 -
* 10 -
* 11 -
* 12 - BASEREG
* 13 - SAVEAREA
* 14 -
* 15 -
*
* R0       EQU   0                  R
R1       EQU   1                   E
R2       EQU   2                    G
R3       EQU   3                     I
R4       EQU   4                      S
R5       EQU   5                       T
R6       EQU   6                        E
R7       EQU   7                         R
R8       EQU   8
R9       EQU   9                         E
R10      EQU   10                            Q
R11      EQU   11                            U
R12      EQU   12                             A
R13      EQU   13                              T
R14      EQU   14                               E
R15      EQU   15                                S
*
* AMUDSNXP DEFINES THE USER VARIABLE USER EXIT PARM BLOCK.
* *
* NOTE:
* *
* YOU MAY NOT MODIFY THE FIELDS IN FRONT OF THE USER AREA.
* *
* YOU MAY NOT MODIFY THE 'STRUCTURE' OF THIS DSECT.
* *
* AMUDSNXP DSECT ,                  PARMS PASSED TO EXIT
* * INPUT AREA
* XPJOBN   DS    CL8                JOBNAME
XPSTEP   DS    CL8                STEPNAME
XPDBNAME DS    CL8                DATABASE NAME
XPSPNAME DS    CL8                SPACE NAME
XPRESUME DS    CL1                RESUME (Y/N)
XPREPLAC DS    CL1                REPLACE (Y/N)
DS    CL2                RESERVED FOR LOADPLUS
XPUSER   DS    CL8                USER ID
XPSSID   DS    CL4                DB2 SUBSYSTEM ID
XPDATE   DS    CL6                UTILITY EXECUTION DATE MMDDYY
**LOADPLUS Assembler user exit**

XPTIME   DS    CL6                UTILITY EXECUTION TIME HHMMSS
XPUTILID DS    CL16               UTILITY ID
XPDATE8  DS    CL8                UTILITY EXECUTION DATE MMDDYYYY
XPGRPNM  DS    CL4                DATA SHARING GROUP NAME
XPVCAT   DS    CL8                VCAT NAME (FROM 1ST PART IF PARTITND)
XPDATEJ  DS    CL7                UTILITY EXECUTION DATE CCYYDDD
DS    CL13               RESERVED FOR LOADPLUS

*.USER WORK AREA

XPUSRWD1 DS    F                  USER WORD 1
XPUSRWD2 DS    F                  USER WORD 2
XPUSRWD3 DS    F                  USER WORD 3
XPUSRWD4 DS    F                  USER WORD 4
XPUSRWD5 DS    F                  USER WORD 5
XPUSRWD6 DS    F                  USER WORD 6
XPUSRWD7 DS    F                  USER WORD 7
XPUSRWD8 DS    F                  USER WORD 8

* OUTPUT AREA

XPUVAREA DS    100CL17            USER VARIABLE AREA
XPUVENT# EQU   100                NUMBER OF ENTRIES IN OUTPUT AREA
XP$      EQU   *-AMUDSNXP

*---------------------------------------------------------------------*
*XPUVARS DEFINES THE OUTPUT AREA OF THE AMUDSNXP DSECT AT LABEL*XPUVAREA. YOU MAY USE THIS DSECT TO EASILY ADDRESS THE TABLE*XPUVAREA. *
*NOTE: THE NUMBER OF ENTRIES IN THE XPUVAREA TABLE MUST NOT EXCEED 100*OR YOU WILL ADDRESS BEYOND THE END OF THE GETMAINED STORAGE*PROVIDED FOR THIS ROUTINE.

* VARIABLE NAME: *

1. MUST BEGIN WITH AN UNDERSCORE CHARACTER C’_’ X’6D’ *
2. MUST BEGIN IN THE LEFTMOST BYTE OF XPUVNAME FIELD *
3. IF LESS THAN 9 BYTES, MUST BE PADDED ON THE RIGHT WITH BLANKS *
4. MAY NOT CONTAIN EMBEDDED BLANKS *
5. NON-BLANK CHARACTERS AFTER THE LEFTMOST BYTE MUST BE NATIONAL CHARACTERS WHICH ARE A THROUGH Z, 0 THROUGH 9, #, @, AND $ *

* VARIABLE DATA: *

1. MUST BEGIN IN THE LEFTMOST BYTE OF XPUVDATA FIELD *
2. IF LESS THAN 8 BYTES, MUST BE PADDED ON THE RIGHT WITH BLANKS *
3. MAY NOT CONTAIN EMBEDDED BLANKS *
4. NON-BLANK CHARACTERS MUST BE 'NATIONAL' CHARACTERS WHICH ARE A THROUGH Z, 0 THROUGH 9, #, @, $, -, AND . *

*---------------------------------------------------------------------*

XPUVARS  DSECT ,                  MAP A VARIABLE ENTRY
XPUVNAME DS    CL9                VARIABLE NAME
XPUVDATA DS    CL8                VARIABLE DATA
XPUVENT$ EQU   *-XPUVNAME         LENGTH OF EACH ENTRY
EJECT

*---------------------------------------------------------------------*

* LOCAL PROGRAM WORK AREA DSECT POINTED TO BY R11
WORKAREA DSECT
SAVE DS 18F LOCAL SAVE AREA
DWORK DS D DOUBLEWORD WORK AREA
CWORK DS CL8 CHARACTER WORK AREA
WRKAREA$ EQU *-WORKAREA

*---------------------------------------------------------------------*
*                                                                     *
*                     P R O G R A M   S T A R T                       *
*                                                                     *
*---------------------------------------------------------------------*
*
AMUEDSNA CSECT
*
*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>><<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*
*                                                                     *
* UNCOMMENT THE NEXT TWO INSTRUCTIONS TO DISABLE THIS EXIT.           *
*                                                                     *
*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>><<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*
*                                                                     *
*        LA    R15,0                    RC=0, GOOD RETURN                *
*        BSM   0,R14                    JUST RETURN TO CALLER          *
*                                                                     *
*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>><<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*
*                                                                     *
* UNCOMMENT THE PREVIOUS TWO INSTRUCTIONS TO DISABLE THIS EXIT.       *
*                                                                     *
*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>><<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*
*                                                                     *
*        STM   R14,R12,12(R13)          SAVE CALLERS REGS              *
*        LR    R12,R15                  R12 IS MY BASEREG               *
*        USING AMUEDSNA,R12             ESTABLISH ADDRESSABLITY        *
*                                                                     *
*        LR    R10,R1                   GET A(PASSED BLOCK)             *
*        USING AMUDESNXP,R10            MAP IT                         *
*                                                                     *
*---------------------------------------------------------------------*
*                                                                     *
* GET SOME WORKING STORAGE                                            *
*                                                                     *
*---------------------------------------------------------------------*
*                                                                     *
*        XR    R11,R11                  ZERO R11                        *
*        GETMAIN RC,LV=WRKAREA$,LOC=BELOW GET WORK AREA                *
*        LTR   R15,R15                  OK?                              *
*        BNZ   DSNXRC08                 ..NO, ERROR                       *
*        LR    R11,R1                   R11 = A(USER WORK AREA)           *
*        ST    R11,XPUSRWD1              SAVE A(USER WORK AREA)            *
*        USING WORKAREA,R11             ADDRESS WORKAREA DSECT          *
*        LA    R15,SAVE                 MY SAVE AREA@                     *
*        ST    R15,.R13                 SAVE IN CALLERS SAVE AREA        *
*        ST    R13,.R15                 SAVE CALLERS SAVE AREA@ IN MINE  *
*        LR    R13,R15                 SET OURS CURRENT                  *
*        EJECT                          *
*                                                                     *
*---------------------------------------------------------------------*
*                                                                     *
* CREATE USER-DEFINED VARIABLES                                     *
*                                                                     *
* THE FOLLOWING USER VARIABLES WILL BE CREATED:                      *
*                                                                     *
*   _JDATE     - JULIAN DATE IN THE FORM OF Dyyddd BUILT FROM THE     *
*                LOADPLUS SUPPLIED VARIABLE &DATE8                    *
*                                                                     *
*   _JCDATE    - JULIAN DATE IN THE FORM OF Dyyyyddd BUILT FROM THE    *
*                LOADPLUS Assembler user exit
LOADPLUS SUPPLIED VARIABLE &DATE8

_UTILPFX - UP TO 8 BYTES OF THE FIRST NODE OF THE UTILITY ID

_UTILISFX - UP TO 8 BYTES OF THE REMAINING PORTION OF THE UTILITY ID

LA R3,XPUVAREA @OUTPUT VARIABLE AREA
USING XPUVARS,R3 ESTABLISH BASE

BUILD _JDATE AND _JCDATE USER VARIABLES

MVC XPUVNAME(XPUVENT$),BLANKS CLEAR ENTRY

MVC XPUVNAME,=CL9'_JDATE' SET VAR NAME

LA R2,MONTHTAB @MONTH TABLE - NO LEAP YEAR
PACK DWORK(8),XPDATE8+4(4) GET YEARS
CVB R4,DWORK INTO R4
LR R7,R4 SAVE A COPY IN R7
SRDL R4,32 PREPARE FOR DIVIDE
D R4,=F'4' DIVIDE BY 4
LTR R4,R4 LEAP YEAR?
BNZ DSNX0110 NO ->
LR R4,R7 GET YEAR AGAIN
SRDL R4,32 PREPARE FOR DIVIDE
D R4,=F'100' DIVIDE BY 100
LTR R4,R4 LEAP YEAR?
BNZ DSNX0100 YES ->
LR R4,R7 GET YEAR AGAIN
SRDL R4,32 PREPARE FOR DIVIDE
D R4,=F'400' DIVIDE BY 400
LTR R4,R4 LEAP YEAR?
BNZ DSNX0110 NO ->

DSNX0100 DS OH

LA R2,LEAPTAB @MONTH TABLE - LEAP YEAR

DSNX0110 DS OH

PACK DWORK(8),XPDATE8(2) GET MONTH
CVB R4,DWORK INTO R4
BCTR R4,0 SUBTRACT 1
SLL R4,1 MULTIPLY BY 2 FOR OFFSET
LH R6,0(R2,R4) GET DAYS FOR PRIOR MONTHS
PACK DWORK(8),XPDATE8+2(2) GET DAYS
CVB R4,DWORK INTO R4
AR R6,R4 TOTAL DAYS IN R6
LR R5,R7 GET YEAR
M R4,=F'1000' SHIFT THE YEAR OVER
AR R5,R6 MAKE IT YYDDD FORMAT
CVD R5,DWORK MAKE IT PACKED
MVC CWORK(L'PL4PAT),PL4PAT GET EDIT PATTERN
ED CWORK(L'PL4PAT),DWORK+4 EDIT YYDDD

MVI XPUVDATA,C'D' SET VAR DATA
MVC XPUVDATA+1(5),CWORK+3 SET VAR DATA

LA R3,XPUVENT$,.R3 @NEXT ENTRY
MVC XPUVNAME(XPUVENT$),BLANKS CLEAR ENTRY
* MVC XPUVNAME,=CL9'JCDATE' SET VAR NAME
* CVD R5,DWORK MAKE IT PACKED
  MVC CWORK(L'PL4PAT),PL4PAT GET EDIT PATTERN
  ED CWORK(L'PL4PAT),DWORK+4 EDIT YYYYDDD
* MVI XPUVDATA,C'D' SET VAR DATA
  MVC XPUVDATA+1(7),CWORK+1 SET VAR DATA
* *---------------------------------------------------------------------*
* BUILD _UTILPFX AND _UTILSFX USER VARIABLES                        *
* *---------------------------------------------------------------------*
* LA R3,XPUVENT$(,R3) @(NEXT ENTRY)
  MVC XPUVNAME(XPUVENT$),BLANKS CLEAR ENTRY
* * MVC XPUVNAME,=CL9'_UTILPFX' SET VAR NAME
* TRT XPUTILID,TRTAB LOOK FOR DELIMITER
  BZ DSNX0200 NONE ->
* LA R2,XPUTILID @(UTIL ID)
  LR R4,R1 SAVE @(DELIMITER)
  SR R1.R2 LENGTH OF FIRST PART
  BZ DSNX0200 NONE ->
  C R1.=F'8' MORE THAN 8 BYTES LONG
  BNH DSNX0130 NO ->
  LA R1,8 MAKE IT 8 BYTES
  DSNX0130 DS 0H
  BCTR R1,0 SUBTRACT 1 FOR EXECUTED MVC
  LA R5,XPUTILID @(UTILITY)
  EX R1,DSNXMVCU MOVE UTILID PREFIX
* * LA R3,XPUVENT$(,R3) @(NEXT ENTRY)
  MVC XPUVNAME(XPUVENT$),BLANKS CLEAR ENTRY
* MVC XPUVNAME,=CL9'_UTILSFX' SET VAR NAME
  LA R5,1(,R4) @(1ST CHAR PAST DELIMITER)
  LA R1,XPUTILID+L'XPUTILID @(END OF UTILID)
  SR R1.R5 LENGTH OF SECOND PART OF UTILID
  BZ DSNXRC00 NONE ->
  C R1.=F'8' MORE THAN 8 BYTES LONG
  BNH DSNX0140 NO ->
  LA R1,8 MAKE IT 8 BYTES
  DSNX0140 DS 0H
  BCTR R1,0 SUBTRACT 1 FOR EXECUTED MVC
  EX R1,DSNXMVCU MOVE UTILID PREFIX
* B DSNXRC00 RETURN RC=0
  DSNX0200 DS 0H
* MVC XPUVDATA,XPUTILID GET FIRST 8 BYTES
* LA R3,XPUVENT$(,R3) @(NEXT ENTRY)
  MVC XPUVNAME(XPUVENT$),BLANKS CLEAR ENTRY
* MVC XPUVNAME,=CL9'_UTILSFX' SET VAR NAME
  MVC XPUVDATA,XPUTILID+8 GET NEXT 8 BYTES
DSNXRC00  DS    0H
   LA    R3,0                     SAVE RETURN CODE
   B     DSNX9000                   GOOD RETURN
DSNXRC08  DS    0H
   LA    R3,8                     SAVE RETURN CODE
                   TERMINATE LOADPLUS EXECUTION
DSNX9000  DS    0H
   LTR   R11,R11                  DID WE GET STORAGE?
   BZ    DSNX9900                  NO ->
   L     R13,SAVE+4
       FREEMAIN RC, LV=WRKAREA$, A=(R11) FREE LOCAL WORK AREA
DSNX9900  DS    0H
   LR    R15,R3                   RESTORE RETURN CODE
   L     R14,12(,R13)
   LM    R0,R12,20(R13)
   BSM   0,R14                    RETURN
EJECT

CONSTANTS

0 1 2 3 4 5 6 7 8 9 A B C D E F
TRTAB   DC   X'00000000000000000000000000000000' 0
   DC   X'00000000000000000000000000000000' 1
   DC   X'00000000000000000000000000000000' 2
   DC   X'00000000000000000000000000000000' 3
   DC   X'FF000000000000000000000000000000' 4 (SP) . + |
   DC   X'FFFF0000000000000000000000000000' 5 : 
   DC   X'FFFF0000000000000000000000000000' 6 - / , _
   DC   X'00000000000000000000000000000000' 7 : =  
   DC   X'00000000000000000000000000000000' 8
   DC   X'00000000000000000000000000000000' 9
   DC   X'00000000000000000000000000000000' A
   DC   X'00000000000000000000000000000000' B
   DC   X'00000000000000000000000000000000' C
   DC   X'00000000000000000000000000000000' D
   DC   X'FFFF0000000000000000000000000000' E \  
   DC   X'00000000000000000000000000000000' F
MONTHTAB  DC   H'0'  JANUARY
LOADPLUS COBOL II and LE COBOL user exit

The topics in this section provides a sample exit parameter record, variable mapping record, and exit that you can use to write a user exit in COBOL or LE COBOL.

COBOL II and LE COBOL exit parameter record

The COBOL exit parameter record contains both input and output fields.

Input fields pass vital information to the user exit, such as the database name and user ID. If you make any modifications to these input fields, LOADPLUS disregards them on return. The output fields pass information about your user variables back to LOADPLUS.

Figure 34: COBOL II and LE COBOL exit parameter record

```cobol
*-----------------------------------------------------------------  
01  LOAD-EXIT-PARMS.  
   05  FIXED-PARM-VALUES.  
      10  EXIT-JOBNAME   PIC X(8).  
      10  EXIT-STEPNAME  PIC X(8). 
*-----------------------------------------------------------------
```
The following table describes the major parameter record fields:

**Table 150: COBOL user exit parameter record fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXIT-JOBNAME</td>
<td>Contains the job name, up to 8 bytes</td>
</tr>
<tr>
<td>EXIT-STEPNAME</td>
<td>Contains the step name, up to 8 bytes</td>
</tr>
<tr>
<td>EXIT-DBNAME</td>
<td>Contains the database name, up to 8 bytes</td>
</tr>
<tr>
<td>EXIT-TSNAME</td>
<td>Contains the name of the table space or index space from the LOAD command, up to 8 bytes</td>
</tr>
<tr>
<td>EXIT-RESUME</td>
<td>Contains an indication of the whether a LOAD RESUME is being performed: Y (yes) or N (no)</td>
</tr>
<tr>
<td>EXIT-REPLACE</td>
<td>Contains an indication of the whether a LOAD REPLACE is being performed: Y (yes) or N (no)</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>EXIT-USERID</td>
<td>Contains the user ID of the user running the LOADPLUS utility, up to 8 bytes</td>
</tr>
<tr>
<td>EXIT-DB2-SSID</td>
<td>Contains the DB2 subsystem ID, 4 bytes</td>
</tr>
<tr>
<td>EXIT-DATE</td>
<td>Contains the date of the execution of the utility, in the format MMDDYY, 6 bytes</td>
</tr>
<tr>
<td>EXIT-TIME</td>
<td>Contains the time of the execution of the utility, in the format HHMMSS, 6 bytes</td>
</tr>
<tr>
<td>EXIT-UTILID-PARM</td>
<td>Contains the utility ID, up to 16 bytes</td>
</tr>
<tr>
<td>EXIT-DATE8</td>
<td>Contains the date of the execution of the utility, in the format MMDDYYYY, 8 bytes</td>
</tr>
<tr>
<td>EXIT-GRPNM</td>
<td>Contains the DB2 data sharing group name&lt;br&gt;In a non-data-sharing environment, the field contains the DB2 subsystem ID.</td>
</tr>
<tr>
<td>EXIT-VCAT</td>
<td>Contains the VCATNAME specified in the DB2 catalog for the table space being loaded, or for the first partition if the table space is partitioned</td>
</tr>
<tr>
<td>EXIT-DATEJ</td>
<td>Contains the Julian date of the execution of the utility, in the format CCYYDDD, 7 bytes</td>
</tr>
<tr>
<td>WORK-AREA-1...WORK-AREA-8</td>
<td>Provides work space, 8 parameters, up to 4 bytes each</td>
</tr>
</tbody>
</table>

**COBOL II and LE COBOL variable mapping record**

The COBOL variable mapping record defines the output area of the exit parameter record.

Using this record allows you to easily address the variable definition table. The number of entries in this table (each entry consists of both a variable name and its value) must not exceed 100.

**Figure 35: COBOL II and LE COBOL variable mapping record**

```
05 USER-DEFINED-VARIABLE-TABLE OCCURS 100 TIMES.
  10 VARIABLE-NAME       PIC X(9).
  10 VARIABLE-VALUE      PIC X(8).
  10 FILLER REDEFINES VARIABLE-VALUE.
    15 VARIABLE-PREFIX    PIC X(1).
    15 VARIABLE-JUL-DATE  PIC X(7).
```

The following table describes the major variable mapping record fields:
Table 151: COBOL user exit variable mapping record fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>USER-DEFINED-VARIABLE-TABLE</td>
<td>The table containing user-defined variable information. See “Rules for the user-defined variables created in LOADPLUS user exits” on page 827 for details about establishing user-defined variables.</td>
</tr>
<tr>
<td>VARIABLE-NAME</td>
<td>The name of the variable that you are defining</td>
</tr>
<tr>
<td>VARIABLE-VALUE</td>
<td>The value of the variable that you are defining</td>
</tr>
</tbody>
</table>

Sample COBOL II and LE COBOL user exit

This topic illustrates a sample user exit written in COBOL.

**Figure 36: Sample COBOL II and LE COBOL user exit**

```
* ALL COBOL MODULES MUST BE COMPILED WITH DATA(31) AND DYNAM TO
* EXECUTE PROPERLY!!!!!!!!
*-----------------------------------------------------------------
IDENTIFICATION DIVISION.
*-----------------------------------------------------------------
PROGRAM-ID. AMUEDSN2.
AUTHOR. BMC SOFTWARE
DATE-WRITTEN. AUGUST 1995.
DATE-COMPILED.
*-----------------------------------------------------------------
* AMUEDSN2 IS A SAMPLE DB2 COBOL II USER EXIT.
* THIS IS A SAMPLE LOADPLUS USER EXIT. THIS EXIT WOULD BE USED
* IN ORDER TO DEFINE USER VARIABLES FOR BUILDING DATASET NAMES
* FOR DYNAMIC WORKFILE ALLOCATION WITH THE DSNPAT KEYWORD.
* AMUEDSN2 IS CALLED ONLY ONCE PER EXECUTION OF AMUUMAIN
* THE MODULE IS CALLED AT THE BEGINNING OF EITHER THE PRELOAD
* PHASE OR THE COMBINED PHASE.
* THIS EXIT WILL ONLY BE INVOKED WHEN IT IS THE VALUE SET FOR
* DSNUEXIT PARM IN AMU$OPTS DURING INSTALLATION OR IN THE
* DSNUEXIT PARAMETER OF THE LOAD COMMAND. THE LOAD COMMAND
* OPTION WILL OVERRIDE THE PARM IN THE AMU$OPTS MACRO.
* PLEASE REVIEW ADDITIONAL DOCUMENTATION IN THE REFERENCE MANUAL
*-----------------------------------------------------------------
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-Control.
DATA DIVISION.
FILE SECTION.
EJECT
*-----------------------------------------------------------------
WORKING-STORAGE SECTION.
*-----------------------------------------------------------------
01 FILLER PIC X(16) VALUE 'WORKING STORAGE '.
```
01 MISCELLANEON.
  05 SAVE-DATA PIC X(8) VALUE SPACES.
  05 DAYS-SUBX PIC S9(3) COMP VALUE ZERO.
  05 SUBX PIC S9(3) COMP VALUE ZERO.
  05 ONE PIC S9(1) COMP-3 VALUE +1.
  05 TWO PIC S9(1) COMP-3 VALUE +2.
  05 FOUR PIC S9(1) COMP-3 VALUE +4.
  05 MAX-LENGTH PIC S9(3) COMP-3 VALUE +16.
  05 NINETEEN PIC S9(3) COMP-3 VALUE +19.
  05 TWENTY PIC S9(3) COMP-3 VALUE +20.
  05 NINETY-FIVE PIC S9(3) COMP-3 VALUE +95.
  05 ONE-HUNDRED PIC S9(3) COMP-3 VALUE +100.
  05 FOUR-HUNDRED PIC S9(3) COMP-3 VALUE +400.
  05 JULIAN-DATE-DESC PIC X(9) VALUE '_JDATE'.
  05 JULIAN-CDATE-DESC PIC X(9) VALUE '_JCDATE'.
  05 UTILITY-PREFIX PIC X(9) VALUE '_UTILPFX'.
  05 UTILITY-SUFFIX PIC X(9) VALUE '_UTILSFX'.
  05 UTILID-PREFIX PIC X(8) VALUE SPACES.
  05 UTILID-SUFFIX PIC X(8) VALUE SPACES.
  05 UTILID-POINTER PIC S9(3) VALUE ZERO.
  05 UTILID-COUNTER PIC S9(3) VALUE ZERO.
  05 UTILID-TALLY PIC S9(3) VALUE ZERO.

01 DATE-WORK-AREA.
  05 CONVERTED-DATE.
    10 DATE-PREFIX PIC X(1) VALUE 'D'.
    10 JULIAN-DATE PIC 9(7) VALUE ZERO.
    10 FILLER REDEFINES JULIAN-CDATE.
      15 JULIAN-CC PIC 9(2).
      15 JULIAN-DATE PIC 9(5).
    15 FILLER REDEFINES JULIAN-DATE.
      20 JULIAN-YY PIC 9(2).
      20 JULIAN-DAYS PIC 9(3).
  05 WORK-YEAR PIC S9(3) COMP-3 VALUE ZERO.
  05 YEAR-ANSWER PIC S9(3) COMP-3 VALUE ZERO.
  05 YEAR-REMAINDER PIC S9(9) COMP-3 VALUE ZERO.
  05 FILLER REDEFINES YEAR-REMAINDER.
    10 YEAR-X PIC X(5).

*------------------------------------------------------------------
* THE FOLLOWING TWO TABLES ARE USED TO CALCULATE THE JULIAN
* DAY DEPENDING ON WHETHER THE YEAR IS A LEAP OR NOT.
*------------------------------------------------------------------

01 NO-LEAP-MONTHS.
  05 JANUARY PIC S9(3) COMP-3 VALUE 00.
  05 FEBRUARY PIC S9(3) COMP-3 VALUE 31.
  05 MARCH PIC S9(3) COMP-3 VALUE 59.
  05 APRIL PIC S9(3) COMP-3 VALUE 90.
  05 MAY PIC S9(3) COMP-3 VALUE 120.
  05 JUNE PIC S9(3) COMP-3 VALUE 151.
  05 JULY PIC S9(3) COMP-3 VALUE 181.
  05 AUGUST PIC S9(3) COMP-3 VALUE 212.
  05 SEPTEMBER PIC S9(3) COMP-3 VALUE 243.
  05 OCTOBER PIC S9(3) COMP-3 VALUE 273.
  05 NOVEMBER PIC S9(3) COMP-3 VALUE 304.
  05 DECEMBER PIC S9(3) COMP-3 VALUE 334.
  01 FILLER REDEFINES NO-LEAP-MONTHS.
    05 MONTH-DAYS PIC S9(3) COMP-3 OCCURS 12 TIMES.

01 LEAP-MONTHS.
  05 LEAP-JANUARY PIC S9(3) COMP-3 VALUE 00.
05 LEAP-FEBRUARY  PIC 9(3) COMP-3 VALUE 31.
05 LEAP-MARCH    PIC 9(3) COMP-3 VALUE 60.
05 LEAP-APRIL    PIC 9(3) COMP-3 VALUE 91.
05 LEAP-MAY      PIC 9(3) COMP-3 VALUE 121.
05 LEAP-JUNE     PIC 9(3) COMP-3 VALUE 152.
05 LEAP-JULY     PIC 9(3) COMP-3 VALUE 182.
05 LEAP-AUGUST   PIC 9(3) COMP-3 VALUE 213.
05 LEAP-SEPTEMBER PIC 9(3) COMP-3 VALUE 244.
05 LEAP-OCTOBER  PIC 9(3) COMP-3 VALUE 274.
05 LEAP-NOVEMBER PIC 9(3) COMP-3 VALUE 305.
05 LEAP-DECEMBER PIC 9(3) COMP-3 VALUE 335.
05 LEAP-MONTH-DAYS PIC 9(3) COMP-3 OCCURS 12 TIMES.

FILLER REDEFINES LEAP-MONTHS.

LINKAGE SECTION.

THE TABLE CAN NOT OCCUR MORE THAN 100 TIME OR A STORAGE
OVERLAY WILL OCCUR.

LOAD-EXIT-PARMS.
05 FIXED-PARM-VALUES.
  10 EXIT-JOBNAME       PIC X(8).
  10 EXIT-STEPNAME      PIC X(8).
  10 EXIT-DNNAME        PIC X(8).
  10 EXIT-DSNAME        PIC X(8).
  10 EXIT-RESUME        PIC X(1).
  10 EXIT-REPLACE       PIC X(1).
  10 EXIT-FILLER1       PIC X(2).
  10 EXIT-USERID        PIC X(8).
  10 EXIT-DB2-SSID      PIC X(4).
  10 EXIT-DATE.
    15 EXIT-MM          PIC 9(2).
    15 EXIT-DD          PIC 9(2).
    15 EXIT-YY          PIC 9(2).
  10 EXIT-TIME          PIC X(6).
  10 EXIT-UTILID-PARM   PIC X(16).
  10 FILLER REDEFINES EXIT-UTILID-PARM.
    15 EXIT-PREFIX      PIC X(8).
    15 EXIT-SUFFIX      PIC X(8).
  10 EXIT-DATE8.
    15 EXIT-DATE8-MM    PIC 9(2).
    15 EXIT-DATE8-DD    PIC 9(2).
    15 EXIT-DATE8-YY    PIC 9(2).
    15 FILLER REDEFINES EXIT-DATE8-YEAR.
      20 EXIT-DATE8-CC   PIC 9(2).
      20 EXIT-DATE8-YY   PIC 9(2).
  10 EXIT-GRPNNM        PIC X(4).
  10 EXIT-VCAT          PIC X(8).
  10 EXIT-DATEJ.
    15 EXIT-DATEJ-MM    PIC 9(4).
    15 EXIT-DATEJ-DD    PIC 9(4).
    15 FILLER REDEFINES EXIT-DATEJ-YEAR.
      20 EXIT-DATEJ-CC   PIC 9(2).
      20 EXIT-DATEJ-YY   PIC 9(2).
      15 EXIT-DATEJ-DDD  PIC 9(3).
  10 EXIT-FILLER2       PIC X(13).
05 WORK-AREA-ADDRESSES.
  10 WORK-AREA-1        PIC 9(4).
  10 WORK-AREA-2        PIC 9(4).
  10 WORK-AREA-3        PIC 9(4).
  10 WORK-AREA-4        PIC 9(4).
  10 WORK-AREA-5        PIC 9(4).
  10 WORK-AREA-6        PIC 9(4).
  10 WORK-AREA-7        PIC 9(4).
  10 WORK-AREA-8        PIC 9(4).
05 USER-DEFINED-VARIABLE-TABLE OCCURS 100 TIMES.
   10 VARIABLE-NAME PIC X(9).
   10 VARIABLE-VALUE PIC X(8).
   10 FILLER REDEFINES VARIABLE-VALUE.
   15 VARIABLE-PREFIX PIC X(1).
   15 VARIABLE-JUL-DATE PIC X(7).

*-----------------------------------------------------------------
PROCEDURE DIVISION USING LOAD-EXIT-PARMS.
*-----------------------------------------------------------------

0000-MAIN.
   MOVE ZERO TO RETURN-CODE.
   PERFORM 1000-PROCESS-DATE.
   GOBACK.

*-----------------------------------------------------------------
* THE DATE IS PASSED IN A MMDDYYYY FORMAT AND CONVERTED TO
* A JULIAN-DATE(WITH NO CENTURY) FORMAT OR A JULIAN-DATE
* (WITH THE CENTURY) FORMAT.
*-----------------------------------------------------------------

1000-PROCESS-DATE.
   MOVE EXIT-DATE8-DD TO JULIAN-DAYS.
   MOVE EXIT-DATE8-MM TO DAYS-SUBX.
   MOVE EXIT-DATE8-CC TO JULIAN-CC.
   MOVE EXIT-DATE8-YY TO JULIAN-YY.
   MOVE EXIT-DATE8-YEAR TO WORK-YEAR.
   DIVIDE WORK-YEAR BY FOUR
       GIVING YEAR-ANSWER
       REMAINDER YEAR-REMAINDER.
   IF YEAR-REMAINDER > ZERO
       THEN
       PERFORM 1100-NO-LEAP-YEAR
   ELSE
       DIVIDE WORK-YEAR BY ONE-HUNDRED
           GIVING YEAR-ANSWER
           REMAINDER YEAR-REMAINDER
       IF YEAR-REMAINDER > ZERO
           THEN
           PERFORM 1200-LEAP-YEAR
       ELSE
       END-IF
       ELSE
       END-IF.

1000-PROCESS-DATE-EXIT.
   EXIT.

1100-NO-LEAP-YEAR.
   ADD MONTH-DAYS(DAYS-SUBX) TO JULIAN-DAYS.
   PERFORM 1300-CENTURY.

1100-NO-LEAP-YEAR-EXIT.
   EXIT.
1200-LEAP-YEAR.
   ADD LEAP-MONTH-DAYS(DAYS-SUBX) TO JULIAN-DAYS.
   PERFORM 1300-CENTURY.

1200-LEAP-YEAR-EXIT.
   EXIT.

1300-CENTURY.
   PERFORM 1400-EDIT-UTILID.

1300-CENTURY-EXIT.
   EXIT.

*-----------------------------------------------------------------
*    THE FULL 16 BYTES OF THE UTILITY ID PARM IS CHECKED FOR
*    A DELIMITER.  IF ONE IS FOUND AFTER THE FIRST 8 BYTES, IT
*    IS INCLUDED IN THE SUFFIX.  IF A DELIMITER IS FOUND IN THE
*    FIRST 8 BYTES, ONLY THOSE CHARACTERS/NUMBERS UP TO THE
*    DELIMITER, WILL BE MOVED INTO THE PREFIX FIELD.
*-----------------------------------------------------------------
*    THE DELIMITER CHARACTER IMMEDIATELY FOLLOWING THE FORWARD
*    SLASH ('/') IS A BROKEN VERTICAL BAR, HEX 6A, AND IS NOT
*    DISPLAYABLE IN BOOKMANAGER
*

1400-EDIT-UTILID.
   MOVE ONE TO UTILID-POINTER, UTILID-TALLY.
   UNSTRING EXIT-UTILID-PARM
      DELIMITED BY ' ' OR :: OR '+' OR '|' OR '::'
      OR '-' OR '/' OR :: OR '|' OR '_' OR ':'
      OR '=' OR '\'
   INTO UTILID-PREFIX
   COUNT IN UTILID-COUNTER
   WITH POINTER UTILID-POINTER.
   IF UTILID-COUNTER = MAX-LENGTH
      THEN
        MOVE EXIT-SUFFIX TO UTILID-SUFFIX
      ELSE
        MOVE UTILID-COUNTER TO UTILID-POINTER
        ADD TWO TO UTILID-POINTER
        UNSTRING EXIT-UTILID-PARM
        INTO UTILID-SUFFIX
        WITH POINTER UTILID-POINTER
      END-IF.
   PERFORM 1500-UTILID-PARMS.

1400-EDIT-UTILID-EXIT.
   EXIT.

1500-UTILID-PARMS.
   MOVE ONE TO SUBX.
   MOVE UTILITY-PREFIX TO VARIABLE-NAME(SUBX).
   MOVE UTILID-PREFIX TO VARIABLE-VALUE(SUBX).
   ADD ONE TO SUBX.
   MOVE UTILITY-SUFFIX TO VARIABLE-NAME(SUBX).
   MOVE UTILID-SUFFIX TO VARIABLE-VALUE(SUBX).
   ADD ONE TO SUBX.
   MOVE JULIAN-DATE-DESC TO VARIABLE-NAME(SUBX).
   MOVE DATE-PREFIX TO VARIABLE-PREFIX(SUBX).
   MOVE JULIAN-DATE TO VARIABLE-JUL-DATE(SUBX).
   ADD ONE TO SUBX.
   MOVE JULIAN-CDATE-DESC TO VARIABLE-NAME(SUBX).
   MOVE CONVERTED-DATE TO VARIABLE-VALUE(SUBX).
LOADPLUS C user exit

The topics in this section provide a sample exit parameter structure, variable mapping structure, and exit that you can use to write a user exit in C.

C exit parameter structure

The C exit parameter structure contains both input and output fields.

Input fields pass information to the user exit, such as the database name and user ID. If you make any modifications to these input fields, LOADPLUS disregards them on return. The output fields pass information about your user variables back to LOADPLUS.

Figure 37: C exit parameter structure

```c
struct amudsnpx {
    char xpjobn[8];     /* jobname                         */
    char xpstep[8];     /* stepname                        */
    char xpdbname[8];   /* database name                   */
    char xpspname[8];   /* space name                      */
    char xpresume[1];   /* resume   (Y or N)               */
    char xpreplace[1];  /* replace  (Y or N)               */
    char xpresrv1[2];   /* reserved for loadplus           */
    char xpuser[8];     /* user id                         */
    char xpssid[4];     /* db2 subsystem id                */
    char xpdobdb1[6];   /* utility execution date mmddyy */
    char xptime[6];     /* utility execution time hhmmss */
    char xputilid[16];  /* utility id                       */
    char xpdobdb2[8];   /* utility execution date mmddyyyy */
    char xgrpmnm[4];    /* data sharing group name        */
    char xvcn[8];      /* vcat name (from 1st part if partitnd)*/
    char xpdobdb3[13]; /* utility execution date ccyyddd */
    int xuswr1;        /* user word 1*/
    int xuswr2;        /* user word 2*/
    int xuswr3;        /* user word 3*/
    int xuswr4;        /* user word 4*/
    int xuswr5;        /* user word 5*/
    int xuswr6;        /* user word 6*/
    int xuswr7;        /* user word 7*/
    int xuswr8;        /* user word 8*/
... 
```

The following table describes the major structure fields:
Table 152: C user exit parameter structure fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xpjobn</td>
<td>Contains the job name, up to 8 bytes</td>
</tr>
<tr>
<td>xpstep</td>
<td>Contains the step name, up to 8 bytes</td>
</tr>
<tr>
<td>xpdbname</td>
<td>Contains the database name, up to 8 bytes</td>
</tr>
<tr>
<td>xpspname</td>
<td>Contains the name of the table space or index space from the LOAD command, up to 8 bytes</td>
</tr>
<tr>
<td>xpresume</td>
<td>Contains an indication of whether a LOAD RESUME is being performed: Y (yes) or N (no)</td>
</tr>
<tr>
<td>xpreplace</td>
<td>Contains an indication of whether a LOAD REPLACE is being performed: Y (yes) or N (no)</td>
</tr>
<tr>
<td>xpuser</td>
<td>Contains the user ID of the user running the LOADPLUS utility, up to 8 bytes</td>
</tr>
<tr>
<td>xpssid</td>
<td>Contains the DB2 subsystem ID, 4 bytes</td>
</tr>
<tr>
<td>xupdate</td>
<td>Contains the date of the execution of the utility, in the format MMDDYY, 6 bytes</td>
</tr>
<tr>
<td>xptime</td>
<td>Contains the time of the execution of the utility, in the format HHMMSS, 6 bytes</td>
</tr>
<tr>
<td>xputilid</td>
<td>Contains the utility ID, up to 16 bytes</td>
</tr>
<tr>
<td>xupdate8</td>
<td>Contains the date of the execution of the utility, in the format MMDDYYYY, 8 bytes</td>
</tr>
</tbody>
</table>
| xpgprpnm  | Contains the DB2 data sharing group name  
In a non-data-sharing environment, the field contains the DB2 subsystem ID. |
| xpvcat    | Contains the VCATNAME specified in the DB2 catalog for the table space being loaded, or for the first partition if the table space is partitioned |
| xupdatej  | Contains the Julian date of the execution of the utility, in the format CCYYDDD, 7 bytes |
| xpusrwd1...xpusrwd8 | Provides work space, 8 parameters, up to 4 bytes each                        |

C exit variable mapping structure

The C exit variable mapping structure defines the output area of the exit parameter structure.
Using this structure allows you to easily address the variable definition table. The number of entries in this table (each entry consists of both a variable name and its value) must not exceed 100.

**Figure 38: C exit variable mapping structure**

```c
struct XPUVAREA {
    char xpuvname[9];
    char xpuvdata[8];
} xpuvars[XPUVENT];
```

The following table describes the major structure fields:

**Table 153: C user exit variable mapping structure fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XPUVAREA</td>
<td>Defines the area that contains user-defined variable information</td>
</tr>
<tr>
<td></td>
<td>See “Rules for the user-defined variables created in LOADPLUS user exits” on page 827 for details about establishing user-defined variables.</td>
</tr>
<tr>
<td>xpuvname</td>
<td>The name of the variable that you are defining</td>
</tr>
<tr>
<td>xpuvdata</td>
<td>The value of the variable that you are defining</td>
</tr>
<tr>
<td>XPUVENT</td>
<td>Equates to the maximum number of entries in the output variable area</td>
</tr>
<tr>
<td></td>
<td>There is a limit of 100 entries.</td>
</tr>
</tbody>
</table>

**Sample C user exit**

This topic illustrates a sample user exit written in C.

**Figure 39: Sample C user exit**

```c
/* DISCLA IMER */
/* */
/* */
/* THIS IS A SAMPLE LOADPLUS USER EXIT. THIS EXIT WOULD BE USED */
/* IN ORDER TO DEFINE USER VARIABLES FOR BUILDING DATASET NAMES FOR */
/* DYNAMIC WORKFILE ALLOCATION WITH THE DSNPAT KEYWORD. */
/* */
/* THIS EXIT WILL ONLY BE INVOKED WHEN IT IS SPECIFICALLY NAMED */
/* IN THE AMUSOPTS DSNUEXIT PARAMETER OR IN THE DSNUEXIT PARAMETER OF */
/* THE LOAD COMMAND. THE SPECIFICATION IN THE LOAD COMMAND WILL */
/* OVERRIDE THE SPECIFICATION IN THE AMUSOPTS MACRO. IF THE NAME */
/* SPECIFIED IS 'NONE' WITHOUT THE SINGLE QUOTE MARKS, THE EXIT WILL */
/* NOT BE INVOKED. */
/* */
/* NOTE: PLEASE REVIEW THE DOCUMENTATION IN THE REFERENCE MANUAL, */
/* AND THE FOLLOWING USAGE NOTES PRIOR TO IMPLEMENTING THIS EXIT. */
/* */
/* PLEASE CALL BMC SOFTWARE PRODUCT SUPPORT WITH ANY QUESTIONS YOU */
```
/* MAY HAVE IN THIS AREA. */
/* PHONE: 1-800-537-1813 */
/* */

** NOTES **

/* */
/* */
/* AMUEDSC IS CALLED ONCE AT THE BEGINNING OF EITHER THE PRELOAD */
/* PHASE OR THE COMBINED PHASE. */
/* */
/* WHEN INVOKED IT IS PASSED THE ADDRESS OF A USER EXIT BLOCK */
/* DESCRIBED BY THE AMUDSNXP STRUCT. */
/* THIS CALL IS FROM ASEMBLER MODULE DYNWDSNX */
/* */
/* UPON EXIT RETRUN CODE FROM THIS EXIT */
/* */
/* 0 GOOD RETURN */
/* */
/* OTHER ERROR RETURN, TERMINATE UTILITY */
/* */
/* AMUDSNXP STRUCT */
/* THIS STRUCT CONTAINS BOTH INPUT */
/* AND OUTPUT FIELDS. INPUT FIELDS ARE PROVIDED TO */
/* PASS VITAL INFORMATION TO THIS EXIT. ANY MODIFI- */
/* CATIONS TO THESE FIELDS WILL BE DISREGARDED UPON */
/* RETURN TO THE CALLER. THE OUTPUT FIELDS ARE */
/* PROVIDED TO ALLOW THE EXIT TO PASS INFORMATION */
/* BACK TO THE CALLER. */

/* */
/* */
/* XPUVARS DEFINES THE OUTPUT AREA OF THE AMUDSNXP STRUCT AT TAG */
/* XPUVAREA. YOU MAY USE THIS STRUCT TO EASILY REFERENCE THE ARRAY */
/* DEFINED AT XPUVAREA. */
/* */
/* NOTE: */
/* */
/* THE NUMBER OF ENTRIES IN THE XPUVAREA ARRAY MUST NOT EXCEED 100 */
/* OR YOU WILL ADDRESS BEYOND THE END OF THE STRUCTURE STORAGE */
/* PROVIDED FOR THIS ROUTINE. */
/* */
/* VARIABLE NAME: */
/* */
/* 1. MUST BEGIN WITH AN UNDERSCORE CHARACTER C ' _ ' X'6D' */
/* 2. MUST BEGIN IN THE LEFTMOST BYTE OF XPUVNAME FIELD */
/* 3. IF LESS THAN 9 BYTES, MUST BE PADDED ON THE RIGHT WITH BLANKS */
/* 4. MAY NOT CONTAIN EMBEDDED BLANKS */
/* 5. NON-BLANK CHARACTERS AFTER THE LEFTMOST BYTE MUST BE NATIONAL */
/* CHARACTERS WHICH ARE A THROUGH Z, 0 THROUGH 9, #, @, AND $ */
/* 6. MUST BE UPPER CASE. */
/* */
/* VARIABLE DATA: */
/* */
/* 1. MUST BEGIN IN THE LEFTMOST BYTE OF XPUVDATA FIELD */
/* 2. IF LESS THAN 8 BYTES, MUST BE PADDED ON THE RIGHT WITH BLANKS */
/* 3. MAY NOT CONTAIN EMBEDDED BLANKS */
/* 4. NON-BLANK CHARACTERS MUST BE 'NATIONAL' CHARACTERS WHICH ARE */
/* A THROUGH Z, 0 THROUGH 9, #, @, $, -, AND . */
/* */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
define XPUVENT 100
* void dynwdsnx(struct XPVAREA*); */
static int monthtab[] = {
    0, /* january */
    31, /* february */
    59, /* march */
    90, /* april */
    120, /* may */
    151, /* june */
    181, /* july */
    212, /* august */
    243, /* september */
    273, /* october */
    304, /* november */
    334 /* december */
};

static int leaptab[] = {
    0, /* january */
    31, /* february */
    60, /* march */
    91, /* april */
    121, /* may */
    152, /* june */
    182, /* july */
    213, /* august */
    244, /* september */
    274, /* october */
    305, /* november */
    335 /* december */
};

struct amudsnpx {
    char xjobn[8]; /* jobname */
    char xstep[8]; /* stepname */
    char xdbname[8]; /* database name */
    char xspname[8]; /* space name */
    char xresume[1]; /* resume (Y or N) */
    char xpreplace[1]; /* replace (Y or N) */
    char xpresrv1[2]; /* reserved for loadplus */
    char xuser[8]; /* user id */
    char xssid[4]; /* db2 subsystem id */
    char xdate[6]; /* utility execution date mmddyy */
    char xptime[6]; /* utility execution time hhmmss */
    char xputilid[16]; /* utility id */
    char xgrpnm[4]; /* data sharing group name */
    char xpvcat[8]; /* vcat name (from 1st part if partitnd) */
    char xdatej[7]; /* utility execution date ccyyddd */
    char xpresrv2[13]; /* reserved for loadplus */
    int xusrwd1; /* user word 1 */
    int xusrwd2; /* user word 2 */
    int xusrwd3; /* user word 3 */
    int xusrwd4; /* user word 4 */
    int xusrwd5; /* user word 5 */
    int xusrwd6; /* user word 6 */
    int xusrwd7; /* user word 7 */
    int xusrwd8; /* user word 8 */
}

struct XPUVAREA {
    char xpuvname[9];
    char xpuvdata[8];
} xpuvars[XPUVENT];

int amuedsc (struct amudsnpx *xparm)
{
    char yr[5];
    char day[4];
    char month[4];
    char wuid[17];
    int i;
    int x;
int l;
int wday;
int wyear;
int iyear;
int imonth;
int totdays;
int xdisable;
int pfxlen;
int sfxstrt;
int sfxlen;
struct XPUVAREA *tp;
tp = xparm->xpuvars;
x = l = wday = wyear = iyear = imonth = totdays = 0;
xdisable = i = pfxlen = sfxstrt = sfxlen = 0;
/*--------------------------------------------------------------------*/
/*                                                                    */
/* to disable this exit set xdisable = 1 */
/*                                                                    */
/* if (xdisable) */
/*     return(0); */
/*                                                                    */
/* CREATE USER-DEFINED VARIABLES */
/* THE FOLLOWING USER VARIABLES WILL BE CREATED: */
/* _JDATE _JCDATE _UTILPFX UTILSFX */
/*--------------------------------------------------------------------*/
/*--------------------------------------------------------------------*/
/* BUILD _JDATE AND _JCDATE USER VARIABLES */
memcpy(month, xparm->xpdate8, 2);
memcpy(day, xparm->xpdate8+2, 2);
memcpy(yr, xparm->xpdate8+4, 4);
sscanf(yr, "%d", &wyear);
sscanf(day, "%d", &wday);
sscanf(month, "%d", &imonth);
imonth = imonth - 1;
if ((wyear % 4 == 0 && wyear % 100 != 0 || wyear % 400 == 0))
totdays = wday + leaptab(abs(imonth));
else
    totdays = wday + monthtab(abs(imonth));
sscanf(yr, "%d", &wyear);
sscanf(day, "%03.03d", totdays);
/*--------------------------------------------------------------------*/
/* clear the user variable area to blanks */
memset(tp->xpuvname, ' ', sizeof(struct XPUVAREA));
/*--------------------------------------------------------------------*/
/* clear the user variable area to blanks */
memset(tp->xpuvname,"_JDATE", 6);
memcpy(tp->xpuvdata, "D", 1);
memcpy(tp->xpuvdata+1, xparm->xpdate8+6, 2);
memcpy(tp->xpuvdata+3, day, 3);
/*--------------------------------------------------------------------*/
/*  sprintf(tp->xpuvdata,"D%-2.2s%3.3s",xparm->xpdate8+6,day); */
/*  do not use this format as the '0' used by sprintf will cause    */
/*  BMC51239I USER VARIABLE '_JDATE' HAS INVALID DATA              */
/*--------------------------------------------------------------------*/
tp++;
/*  clear the user variable area to blanks                          */
/*--------------------------------------------------------------------*/
memset(tp->xpuvname, ' ', sizeof(struct XPUVAREA));
memcpy(tp->xpuvname, "_JCDATE", 7);
memcpy(tp->xpuvdata, "D", 1);
memcpy(tp->xpuvdata+1, yr, 4);
memcpy(tp->xpuvdata+5, day, 3);
/*--------------------------------------------------------------------*/
/*  sprintf(tp->xpuvdata,"D%-4.4s%-3.3s",yr,day);                   */
/*  do not use this format as the '\0' used by sprintf will cause   */
/*  BMC51239I USER VARIABLE '_JCDATE' HAS INVALID DATA               */
/*--------------------------------------------------------------------*/
/*                                                                    */
/* build _utilpfx and _utilsfx user variables                         */
/*                                                                    */
/*                                                                    */
tp++;
sprintf(wuid, "%16.16s", xparm->xputilid);
/*--------------------------------------------------------------------*/
/*  get the length of xputilid                                       */
/*  find the first delimiter character                               */
/*  limit _UTILPFX and UTILSFX to 8 characters - DSN node limit       */
/*--------------------------------------------------------------------*/
for (i=0; wuid[i] != '\0'; i++)
{
  switch (wuid[i])
  {
    case ' ':
    case '.':
    case '+':
    case '|':
    case ';':
    case '-':
    case '/':
    case '\':
    case ',':
    case ':':
    case '=':
    case '\\':
      if (pfxlen == 0)
      {
        pfxlen = i;
        sfxstrt = i + 1;
      }
      break;
    default:
      break;
  }
  if (pfxlen == 0) /* no delimiter found */
  {
    pfxlen = 8;
    sfxstrt = 8;
  }
  sfxlen = i - sfxstrt;
  if (pfxlen > 8)
LOADPLUS LE C user exit

The topics in this section provide a sample exit parameter structure, variable mapping structure, and exit that you can use to write a user exit in LE C.

LE C exit parameter structure

The LE C exit parameter structure contains both input and output fields.

Input fields pass vital information to the user exit, such as the database name and user ID. If you make any modifications to these input fields, LOADPLUS disregards them on return. The output fields pass information about your user variables back to LOADPLUS.

Figure 40: LE C exit parameter structure

```c
struct amudsnxp {
    char xpjobn[8];       /* jobname */
    char xpstep[8];       /* stepname */
    char xpdbname[8];     /* database name */
    char xpspname[8];     /* space name */
    char xpresume[1];     /* resume (Y or N) */
    char xpreplace[1];    /* replace (Y or N) */
    char xpresrv1[2];     /* reserved for loadplus */
    char xpuser[8];       /* user id */
    char xpssid[4];       /* db2 subsystem id */
};
```
The following table describes the major structure fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xpjobn</td>
<td>Contains the job name, up to 8 bytes</td>
</tr>
<tr>
<td>xpstep</td>
<td>Contains the step name, up to 8 bytes</td>
</tr>
<tr>
<td>xpdbname</td>
<td>Contains the database name, up to 8 bytes</td>
</tr>
<tr>
<td>xpspname</td>
<td>Contains the name of the table space or index space from the LOAD command, up to 8 bytes</td>
</tr>
<tr>
<td>xpresume</td>
<td>Contains an indication of whether a LOAD RESUME is being performed: Y (yes) or N (no)</td>
</tr>
<tr>
<td>xpreplace</td>
<td>Contains an indication of whether a LOAD REPLACE is being performed: Y (yes) or N (no)</td>
</tr>
<tr>
<td>xpuser</td>
<td>Contains the user ID of the user running the LOADPLUS utility, up to 8 bytes</td>
</tr>
<tr>
<td>xpssid</td>
<td>Contains the DB2 subsystem ID, 4 bytes</td>
</tr>
<tr>
<td>xpdate</td>
<td>Contains the date of the execution of the utility, in the format MMDDYY, 6 bytes</td>
</tr>
<tr>
<td>xptime</td>
<td>Contains the time of the execution of the utility, in the format HHMMSS, 6 bytes</td>
</tr>
<tr>
<td>xputilid</td>
<td>Contains the utility ID, up to 16 bytes</td>
</tr>
<tr>
<td>xpdate8</td>
<td>Contains the date of the execution of the utility, in the format MMDDYYYY, 8 bytes</td>
</tr>
<tr>
<td>xpggrpm</td>
<td>Contains the DB2 data sharing group name</td>
</tr>
<tr>
<td></td>
<td>In a non-data-sharing environment, the field contains the DB2 subsystem ID.</td>
</tr>
</tbody>
</table>
**LE C exit variable mapping structure**

The LE C exit variable mapping structure defines the output area of the exit parameter structure.

Using this structure allows you to easily address the variable definition table. The number of entries in this table (each entry consists of both a variable name and its value) must not exceed 100.

*Figure 41: LE C exit variable mapping structure*

```c
struct XPUVAREA {
    char xpuvname[9];
    char xpuvdata[8];
} xpuvars[XPUVENT];
```

The following table describes the major structure fields:

**Table 155: LE C variable mapping structure fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XPUVAREA</td>
<td>Defines the area that contains user-defined variable information</td>
</tr>
<tr>
<td></td>
<td>See “Rules for the user-defined variables created in LOADPLUS user exits” on page 827 for details about establishing user-defined variables.</td>
</tr>
<tr>
<td>xpuvname</td>
<td>The name of the variable that you are defining</td>
</tr>
<tr>
<td>xpuvdata</td>
<td>The value of the variable that you are defining</td>
</tr>
<tr>
<td>XPUVENT</td>
<td>Equates to the maximum number of entries in the output variable area</td>
</tr>
<tr>
<td></td>
<td>There is a limit of 100 entries.</td>
</tr>
</tbody>
</table>
Sample LE C user exit

This topic illustrates a sample user exit written in LE C.

Figure 42: Sample LE C user exit

/*
  DISCLAIMER
*/
/*
* THIS IS A SAMPLE LOADPLUS USER EXIT. THIS EXIT WOULD BE USED
* IN ORDER TO DEFINE USER VARIABLES FOR BUILDING DATASET NAMES FOR
* DYNAMIC WORKFILE ALLOCATION WITH THE DSNPAT KEYWORD AND FOR
* SHRLEVEL CHANGE WITH THE SPIILDSNPAT KEYWORD.
* THIS EXIT WILL ONLY BE INVOKED WHEN IT IS SPECIFICALLY NAMED
* IN THE AMU$OPTS DSNUEXIT PARAMETER OR IN THE DSNUEXIT PARAMETER OF
* THE LOAD COMMAND. THE SPECIFICATION IN THE LOAD COMMAND WILL
* OVERRIDE THE SPECIFICATION IN THE AMU$OPTS MACRO. IF THE NAME
* SPECIFIED IS 'NONE' WITHOUT THE SINGLE QUOTE MARKS, THE EXIT WILL
* NOT BE INVOKED.
* NOTE: PLEASE REVIEW THE DOCUMENTATION IN THE REFERENCE MANUAL,
* AND THE FOLLOWING USAGE NOTES PRIOR TO IMPLEMENTING THIS EXIT.
* PLEASE CALL BMC SOFTWARE PRODUCT SUPPORT WITH ANY QUESTIONS YOU
* MAY HAVE IN THIS AREA.
  PHONE: 1-800-537-1813
*/

NOTES

AMUEDSL IS CALLED ONCE AT THE BEGINNING OF EITHER THE PRELOAD
PHASE OR THE COMBINED PHASE.
WHEN INVOKE IT IS PASSED THE ADDRESS OF A USER EXIT BLOCK
DESCRIBED BY THE AMUDSNXP STRUCT.
THIS CALL IS FROM ASEMBLER MODULE DYNWDSNX
UPON EXIT RETRUN CODE FROM THIS EXIT
0   GOOD RETURN
OTHER ERROR RETURN, TERMINATE UTILITY
AMUDSNXP STRUCT THIS STRUCT CONTAINS BOTH INPUT
AND OUTPUT FIELDS. INPUT FIELDS ARE PROVIDED TO
PASS VITAL INFORMATION TO THIS EXIT. ANY MODIFI-
CATIONS TO THESE FIELDS WILL BE DISREGARDED UPON
RETURN TO THE CALLER. THE OUTPUT FIELDS ARE
PROVIDED TO ALLOW THE EXIT TO PASS INFORMATION
BACK TO THE CALLER.

XPUVARS DEFINES THE OUTPUT AREA OF THE AMUDSNXP STRUCT AT TAG
XPUVAREA. YOU MAY USE THIS STRUCT TO EASILY REFERENCE THE ARRAY
DEFINED AT XPUVAREA.

NOTE:
THE NUMBER OF ENTRIES IN THE XPUVAREA ARRAY MUST NOT EXCEED 100

Appendix D  LOADPLUS user exits 857
/* OR YOU WILL ADDRESS BEYOND THE END OF THE STRUCTURE STORAGE */
/* PROVIDED FOR THIS ROUTINE. */
/* */
/* VARIABLE NAME: */
/* */
/* 1. MUST BEGIN WITH AN UNDERSCORE CHARACTER C’_ X’6D’ */
/* 2. MUST BEGIN IN THE LEFTMOST BYTE OF XPUVNAME FIELD */
/* 3. IF LESS THAN 9 BYTES, MUST BE PADDED ON THE RIGHT WITH BLANKS */
/* 4. MAY NOT CONTAIN EMBEDDED BLANKS */
/* 5. NON-BLANK CHARACTERS AFTER THE LEFTMOST BYTES MUST BE NATIONAL */
/* 6. MUST BE UPPER CASE. */
/* */
/* VARIABLE DATA: */
/* */
/* 1. MUST BEGIN IN THE LEFTMOST BYTE OF XPUDATA FIELD */
/* 2. IF LESS THAN 8 BYTES, MUST BE PADDED ON THE RIGHT WITH BLANKS */
/* 3. MAY NOT CONTAIN EMBEDDED BLANKS */
/* 4. NON-BLANK CHARACTERS MUST BE ’NATIONAL’ CHARACTERS WHICH ARE */
/* A THROUGH Z, 0 THROUGH 9, #, @, $, -, AND . */
/* */

#pragma runopts(PLIST(HOST))
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
define XPUVENT 100
static int monthtab[] = {
    0,          /* january */
    31,         /* february */
    59,         /* march */
    90,         /* april */
    120,        /* may */
    151,        /* june */
    181,        /* july */
    212,        /* august */
    243,        /* september */
    273,        /* october */
    304,        /* november */
    334         /* december */
};
static int leaptab[] = {
    0,          /* january */
    31,         /* february */
    60,         /* march */
    91,         /* april */
    121,        /* may */
    152,        /* june */
    182,        /* july */
    213,        /* august */
    244,        /* september */
    274,        /* october */
    305,        /* november */
    335         /* december */
};
struct amudsnxp {
    char xpjobn[8];  /* jobname */
    char xpstep[8];  /* stepname */
    char xpdbname[8];  /* database name */
    char xpspname[8];  /* space name */
    char xpresume[1];  /* resume (Y or N) */
    char xpreplace[1];  /* replace (Y or N) */
    char xpresrv1[2];  /* reserved for loadplus */
    char xpuser[8];  /* user id */
    char xpssid[4];  /* db2 subsystem id */
    char xdate[6];  /* utility execution date mmddyy */
    char xtime[6];  /* utility execution time hhmmss */
};
char xputilid[16]; /* utility id */
char xdate8[8]; /* utility execution date mmddyyyy */
char xgrnm[4]; /* data sharing group name */
char xpvc[8]; /* vcat name (from 1st part if partitnd) */
char xdate7[7]; /* utility execution date ccyyddd */
char xresrv2[13]; /* reserved for load plus */
int xpusrwd1; /* user word 1*/
int xpusrwd2; /* user word 2*/
int xpusrwd3; /* user word 3*/
int xpusrwd4; /* user word 4*/
int xpusrwd5; /* user word 5*/
int xpusrwd6; /* user word 6*/
int xpusrwd7; /* user word 7*/
int xpusrwd8; /* user word 8*/
struct XPUVAREA {
    char xpuvname[9];
    char xpuvdata[8];
} xpuvars[XPUVENT];

int main (int argc, char *argv[])
{
    char yr[5];
    char day[4];
    char month[4];
    char wuid[17];
    int i;
    int x;
    int l;
    int wday;
    int wyear;
    int iyear;
    int imonth;
    int totdays;
    int xdisable;
    int pfxlen;
    int sfxstrt;
    int sfxlen;

    struct XPUVAREA *tp;
    struct amudsnxp *xparm;
    xparm = (void *)argv[1];
    tp = xparm->xpuvars;
    x = l = wday = wyear = iyear = imonth = totdays = 0;
    xdisable = i = pfxlen = sfxstrt = sfxlen = 0;

    /*--------------------------------------------------------------------*/
    /*                                                                    */
    /* to disable this exit set xdisable = 1 */
    /*                                                                    */
    /*--------------------------------------------------------------------*/
    if (xdisable)
        return(0);

    /*--------------------------------------------------------------------*/
    /* CREATE USER-DEFINED VARIABLES */
    /*                                                                    */
    /* THE FOLLOWING USER VARIABLES WILL BE CREATED: */
    /*--------------------------------------------------------------------*/
    /* _JDATE - JULIAN DATE IN THE FORM OF DYYDDD BUILT FROM THE */
    /* LOADPLUS SUPPLIED VARIABLE &DATE8 */
    /*--------------------------------------------------------------------*/
    /* _JCDATE - JULIAN DATE IN THE FORM OF DYYYYDDD BUILT FROM THE */
    /* LOADPLUS SUPPLIED VARIABLE &DATE8 */
    /*--------------------------------------------------------------------*/
    /* UTILPFX - UP TO 8 BYTES OF THE FIRST NODE OF THE UTILITY ID */
    /*--------------------------------------------------------------------*/
    /* UTILSFX - UP TO 8 BYTES OF THE REMAINING PORTION OF THE */
    /*--------------------------------------------------------------------*/
memcpy(month, xparm->xpdate8, 2);
memcpy(day, xparm->xpdate8+2, 2);
memcpy(yr, xparm->xpdate8+4, 4);
sscanf(yr, "%d", &wyear);
sscanf(day, "%d", &wday);
sscanf(month, "%d", &imonth);
imonth = imonth - 1;
if (wyear % 4 == 0 && wyear % 100 != 0 || wyear % 400 == 0)
  totdays = wday + leaptab[imonth];
else
  totdays = wday + monthtab[imonth];
sscanf(yr, "%d", &wyear);
sscanf(day, "%03.03d", totdays);
memcpy(tp->xpuvname, "_JDATE", 6);
memcpy(tp->xpuvdata, "D", 1);
memcpy(tp->xpuvdata+1, xparm->xpdate8+6, 2);
memcpy(tp->xpuvdata+3, day, 3);
sscanf(tp->xpuvdata, "D%-2.2s%3.3s", xparm->xpdate8+6, day);
/* do not use this format as the '"0' used by sscanf will cause */
/* BMC51239I USER VARIABLE '_JDATE' HAS INVALID DATA */
/* */
/* */
/* build_utilpfx and _utilsfx user variables */
/* */
/* */
/* */
/* for (i=0; wuid[i] != '\0'; i++)
{ switch (wuid[i])
}
case '.':
case '+':
case '|':
case ';':
case '-':
case '/':
case '':
case ',':
case '_':
case ':':
case '=':  
case '\':
if (pfxlen == 0)
{  
pfxlen = i;
sfxstrt = i + 1;
}
break;
default:
break;
}
if (pfxlen == 0)              /* no delimiter found */
{
    pfxlen = 8;
sfxstrt = 8;
}
sfxlen = i - sfxstrt;
if (pfxlen > 8)
pfxlen = 8;
if (sfxlen > 8)
sfxlen = 8;
/*--------------------------------------------------------------------*/
/*   clear the user variable area to blanks                           */
/*--------------------------------------------------------------------*/
memset(tp->xpuvname,' ', sizeof(struct XPUVAREA));
memcpy(tp->xpuvname, "_UTILPFX", 8);
memcpy(tp->xpuvdata, xparm->xputilid, pfxlen);
/*--------------------------------------------------------------------*/
/*  sprintf(tp->xpuvdata, "%-8.8s", xparm->xputilid);               */
/*  do not use this format as the '\0' used by sprintf will cause     */
/*  BMC51239I USER VARIABLE '_UTILPFX' HAS INVALID DATA               */
/*--------------------------------------------------------------------*/
return (0);
Index

Symbols

(start\end)
    used in predicate 316
    used with CONTINUEIF option 243
    used with POSITION option for field specification 324
&DATE variable 414, 762
&DATEDJ variable 414, 762
&DB variable 414, 762
&DDNAME variable 414, 762
&GRPNM variable 414, 762
&JDATE variable 414, 762
&JOBNAME variable 414, 762
&PART variable 414, 762
&REPLACE variable 414, 762
&RESUME variable 414, 762
&SSID variable 414, 762
&STEPNAME variable 414, 762
&TIME variable 414, 762
&TIME4 variable 414, 762
&TS variable 414, 762
&UID variable 414, 762
&USERID variable 414, 762
&UTIL variable 414, 762
&UTILID variable 414, 762
&UTILPFX variable 414, 762
&UTILSF variable 414, 762
&VCAT variable 414, 762
$532SOPT configuration job 703
$532SOPT job 703
$730DOP2 configuration job 703
$730DOP2 job 703
$AMUDYNA installation options 760
$SORT Parm DD statement 465

A
    abends
        excluding from a system dump 712
    JES3 45, 464
    preventing 55
    restarting after x37 467
    system dump 712
    using ANALYZE statistics to avoid 653
    when specifying RETCODE option 208
    x37 467
    above-the-bar storage 55
    access, controlling
        in LOADPLUS 51
    access, shared 798
    accessibility of objects
        compared to IBM DB2 LOAD 33
        LOADPLUS features 28
        recommendations 688
        SHRLEVEL option 87, 189, 712
    ACF2. See CA ACF2
    ACHKP (auxiliary CHECK pending) status 71
    activating dynamic allocation
        data sets other than SYSREC 398, 762
        SORTWK data sets 446
        SYSREC data sets 211
    ACTIVE option
        command 398
        examples
            basic 562
            command overrides 569
            copy data sets 626
            with AUTOENUMROWS 602
        installation 762
        restart considerations 84
    advisory REORG pending (AREO*) status 74
    advisory restart pending (AREST) status 71
    agents, High-speed Apply 190, 712
    ALL keyword
of DISCARDS option 221
of FORCE option 279, 712
of REGISTER option 294
allocating data sets 432
  copy 436
discard (SYSDISC) 450
disposition parameter (DISP)
  copy data sets 436
effect on DELETFILES option 250
effect on DELFILES option 712
  restart considerations 467
SORTOUT data sets 467
SYSDISC data sets 450
SYSERR data sets 467
SYSUT1 data sets 467
temporary data sets 85
dynamically 78, 393
error (SYSERR) 453
estimating size with ANALYZE option 258,
  465, 653
  index (SYSUT1) 459
input (SYSREC) 211, 456
load (SORTOUT) 443
messages from ANALYZE phase 653
  multiple 670
reallocating during a job 258
  restart considerations 467
sort (SORTWK) 446
  SORTOUT 443
SORTWK 446
SYSDISC 450
SYSERR 453
SYSREC 211, 456
SYSUT1 459
using DD statements 432
  See also individual data set types
allocation-related messages, from ANALYZE phase 653
ALLTYPES keyword of SKIPFIELDS option 304
ALTER authority 51
ALTER privileges 50, 51, 712
ampersand, in DSNPAT installation option 762
AMUSOPTS installation options module 703
ANALYZE option
allocation set allocation estimates
  copy data sets 436
discard (SYSDISC) 450
derror (SYSERR) 453
  index (SYSUT1) 459
load (SORTOUT) 443
  sort work (SORTWK) 446
description 258
determining option to use 653
dynamic allocation considerations 258, 465
  example 538
  of PRELOAD option 208, 443, 456
PAUSE keyword 258, 538
  syntax 258
ANALYZE phase
  allocation-related messages 653
data set allocation estimates 465
  pausing and continuing after analysis 258, 538
  primary functions 38
  tuning considerations 690
ANYDISCARDS keyword of PRELOAD PAUSE
  option 208
APCOLLECTION option
command, of SQLAPPLY option 190
  installation 712
APCOMMIT option
command, of SQLAPPLY option 190
  installation 712
  performance considerations 648, 678
APDOPTS option 712
APF authorization requirements 431
APMAXAGENTS option of SQLAPPLY option 190,
  648, 678
APMULTIROW option
command, of SQLAPPLY option 190
  installation 712
APMXAGNT option 712
APOWNER option
command, of SQLAPPLY option 190
  installation 712
application defaults module 427
application-period temporal tables 185, 712
applied fixes, report of 427
Apply Plus. See High-speed Apply Engine
APRETLIM option 712
AP RETRYLIM option of SQLAPPLY option 190
AP RETRYVAL option of SQLAPPLY option 190
AP RETVAL option 712
APT DUMP data set 435
APTGRANT member of DBSAMP 51
ARCHFORMAT option of REORG PLUS 197
archive data set of REORG PLUS, loading data from 197
AREO* (advisory REORG pending) status 74
AREST (advisory restart pending) status 71
ASCII
  date exit routine 668
  option 204
  time exit routine 668
ASM keyword of DSNUEXIT option 395, 712
assembler, for user exits 395, 712, 828
assigning indexes to data sets 459
assigning partitions for copies 299, 712
ASSOCIATE option of ORDER YES option 225
ASUSRPT data set 289, 436
authorization ID 190
authorization mechanisms, description 50
authorizations
  ALTER 51
  ALTER privileges 50, 51, 712
  APF authorization 431
  BINDADD 51
  compared to IBM DB2 LOAD 33
  CONTROL 51
  CREATE 51
  data set 50
  DB2 and data set
    LOADPLUS 51
  DISPLAY 50
  for canceling threads 50
  INSERT 51
  INSTALL SYSADM authority 51, 712
  LOADPLUS 51
  mechanisms, description 51
  OPNDB2ID option 50
  RACF 50
    LOADPLUS 51
  READ privileges 50, 51
  required for checking referential constraints 51
  required for DEFINE NO data sets 51
  required for loading tables with identity columns 50, 51
  required for staging data sets 50
  required to run LOADPLUS 50
  required to run SQLAPPLY load 51
  system 50, 51
  UPDATE 51
  verifying 51
AUTO keyword
  of ENUMROWS option 231, 648, 659
  of VOLCNT option 408, 762
AUTO ENUMROWS option
  description 712
  examples 538, 562
  overriding, examples 569, 588
  tuning considerations 648, 659
automating file size estimation 231, 712
auxiliary CHECK pending (ACHKP) status 71
auxiliary indexes 206
auxiliary warning (AUXW) status 69, 71
AUXW (auxiliary warning) status 69, 71
availability of objects
  compared to IBM DB2 LOAD 33
  LOADPLUS features 28
  recommendations 688
  SHRLEVEL option 87, 189, 712
available pages 237, 712
AVAILPAGEPCT option 237, 712
average row length, determining 258, 653
AVGSIZE keyword
  of BLOB data type 352
  of BLOBF subtype 338, 340
  of CLOB data type 353
  of CLOBF subtype 338, 340
  of DBCLOB data type 354
  of DBCLOBF subtype 338, 340
  of XML data type 355
AVGVOLSP option 409, 762

B
B37 abends
  SYSDISC data sets 467
  SYSERR data sets 467
backing up BMC tables 789
base tables that participate in a clone relationship 253
BASIC keyword
  of DSNTYPE option 404
  BASIC keyword of DSNTYPE option 762
batch pipe data
  considerations 139
  discards 450
  with INDSN option 211
BatchPipes input 139, 211, 450
BEGINNING keyword of LONGNAMETRUNC option 202, 712
benefits of dynamic data set allocation 78
benefits of LOADPLUS 25
BFP format 347
BIGINT data type 345
binary data
  BINARY data type 348
  BINARY VARYING data type 348
  BLOBF data subtype 338, 340
  conversion 358
  translation 383
  TRIM option 348
  TRUNCATE option 348
  VALUE option 324
  VARBINARY data type 348
XML 338, 340, 355
BINARY data type 348
BINARY VARYING data type 348
BINARYXML option
  of CHAR data type 338
  of VARCHAR data type 340
  of XML data type 355
BINDADD privileges 51
binding packages and plans for High-speed Apply request 190, 712
bitemporal tables. See temporal tables
BLOB data type 352
BLOBF keyword of character data types 338, 340, 712
block size
  greater than 32 KB 656, 712
  of SYSREC data sets 656, 712
  optimal for copy data sets 436
  specifying input data sets based on 456
BMC keyword
  of FORMAT option 197
  of NLPCTFREE option 257
  of RULES option 712
BMC utilities
  displaying status 791
  running concurrently 66, 804
  terminating 791
BMCCPYnn data sets 436
BMCCPZnn data sets 436
BMCDICT table
  considerations 141, 793
  contents 792
  maintaining 793
  with large number of partitions 107
BMCFORCE data set 436
BMCHIST installation option, BMCHIST table 794
BMCHIST table
  backing up 789
  contents 794
  COPY PLUS considerations 796
  maintenance 797
  querying 791
  RECOVER PLUS considerations 796
  when updated 38
BMCLGRNX table 797
BMCPSWD library 289, 431
BMCRCYnn data sets 436
BMCRCZnn data sets 436
BMCSORT
  AVAILPAGEPCT option 237
  controlling 665
  data set that contains messages 464
  description 47
  MAXSORTMEMORY option 242, 712
  memory usage 665
  MINSORTMEMORY option 241, 712
  performance tuning 665
  SHORTMEMORY option 239, 712
  SMCORE option 712
  SORTDEVT option 234, 712
  SORTNUM option 235, 712
  TOTALPAGEPCT option 238, 712
  version requirement 49
  when invoked 691
BMCSORT installation option 783
BMCSTATS option
  description 289
  examples 499, 562
  with SQLAPPLY 97
BMCSTATS tables 38, 289
BMCSYNC table
  backing up 789
  cleaning up RECOVER PLUS UNLOADKEYS 803
  considerations 801
  contents 798
  LOB data considerations 801
  maintaining 803
  purpose 33
  running utilities concurrently 66, 804
  serializing 712
  when updated 38
  with large number of partitions 107
  with partition-by-growth table spaces 110
XML data considerations 801
BMCTRANS table 806
BMCUNLOAD keyword of FORMAT option 197, 588
BMCUTIL table
  backing up 789
  contents 808
  maintaining 811
  serializing 712
  when updated 38
BMCXCOPY table
  backing up 789
  contents 811
  maintaining 818
  querying 791
buffer usage, controlling
  copy data sets 656, 712
  input data sets 656, 712
  load data sets 656, 712
  performance considerations 656
  work data sets 656, 712
BUILD keyword of INDEX option 185
building dictionaries
  command options 187, 306
  installation option 712
  when built 141
building indexes 185, 693
building LOADPLUS jobs 425
BUSINESS TIME WITHOUT OVERLAPS index
  definition 185, 712
BYCLUSTERKEY keyword of ASSOCIATE option 225
BYTABLE keyword of ASSOCIATE option 225

C
C language, for user exits 395, 712, 847
C option 395, 712
CA ACF2 security product 51
CA Top Secret security product 51
caches
  for identity columns 111, 273
  SUF 147
canceling a job 477
canceling threads
authorizations required 50
FORCE option 279, 712
FORCE_AT option 712
FORCE_RPT option 712
reporting 279, 436, 712
specifying when to cancel 279, 712
cardinality, determining 258, 653
CATALOG MANAGER
  dynamic data set allocation considerations 80
CBUFFS option 712
CCSID option
  description 204
  examples 607, 632
  performance considerations 648
  translation processing description 383
CCSID, specified at column level 57, 712
CENTURY option
  command 272
  example 538
  installation 712
  of DATE data type 349
  of TIMESTAMP data type 350
CHANGE keyword of SHRLEVEL option 190, 712
CHANGE MANAGER
  deleting work files, considerations 712
  dynamic data set allocation considerations 80
  dynamic data set allocation, considerations 398
  ROWID columns, considerations 134
changes to the product 16
CHAR data type 338
character conversions (translation), supported 383
character data
  BLOBF keyword 338, 340
  CHAR data type 338
  CLOBF keyword 338, 340
  collating sequence 225
  conversions 358, 376
  DBCLOBF keyword 338, 340
  MIXED keyword 338
  translation 383
  TRIM option 338, 340
  TRUNCATE option 338, 340
  VARCHAR data type 340
characters, delimiting, for CSV format 199
check constraints
checking 263, 712
CHEKPEND option 712
compared to IBM DB2 LOAD 33
discards due to 221
ENFORCE option 263, 712
element 557
how LOADPLUS handles 147, 148
with DEFINE NO objects 148
with DSNUTILB load 148
with FORMAT BMCUNLOAD option 137
with identity columns 113
with SQLAPPLY option 148
CHECK CONSTRAINTS keyword of ENFORCE option 263
CHECK keyword of ENFORCE option 712
CHECK pending (CHKP) status
resetting 74
setting 267, 712
when LOADPLUS sets 150
with LPL or WEPR status 69
CHECK utility after load jobs 688
checking constraints
check 148, 263, 712
informational 146, 263, 712
referential 146, 263, 712
checking uniqueness in indexes 244
CHECKPEND command option 97, 267
CHEKPEND installation option 712
child table spaces, how constraint checking affects status 150
CHKC keyword of DISCARDS option 221
CHKP (CHECK pending) status
resetting 74
setting 267, 712
when LOADPLUS sets 150
with LPL or WEPR 69
classes, SMS 762
cleaning up unused data sets 246, 645
CLOB data type 353
CLOBF keyword of character data types 338, 340, 712
close tables
exchange process for 256
FASTSWITCH restriction 253
statistics restriction 289, 291
CLUSTER keyword of UNIQUECHECK option 244
clustering indexes
ordering by 225
when built 670
clustering key, ordering by 225
CNTLCARDS option (in UNLOAD PLUS) 588
COBOL II
indicating on command 395
indicating on installation option 712
sample user exit 839
COBOL2 keyword of DSNUEXIT option 395, 712
codes, return
designating
on discards 223, 712
on reaching discard limit 224, 712
on zero rows loaded 225, 712
pause on discards 208, 712
determining final 158
differences with IBM DB2 LOAD 33
from user exits 827
hierarchy 158
collating sequence for character data 225
collection ID, High-speed Apply Engine 190, 712
column definitions when using FORMAT
BMCUNLOAD 134
columns
defined with an XML schema 57
defined with AS TRANSACTION START ID 306
defined with CCSID specification 57, 712
document ID 115
identity. See identity columns
row change timestamp 134, 324
system period start and end 138, 306, 324
transaction start 138, 324
COMBINED phase
invoking BMCSORT 695, 698
performance considerations 663
primary functions 38
processes 695, 698
recovery 478
resource allocation diagram 695, 698
restart considerations 467
specifying 208, 712
comma as separator 316
comma-delimited data. See CSV data
coma-separated-value data. See CSV data
command constants 383
command options
configuration jobs 703
constants
  in check constraints 148
  in predicates 316
  in VALUE option 324
CONSTRAINTS keyword of ENFORCE option 263, 712
constraints, check
  checking 263, 712
  CHEKPEND option 712
  compared to IBM DB2 LOAD 33
discards due to 221
ENFORCE option 263, 712
example 557
how LOADPLUS handles 147, 148
with DEFINE NO objects 148
with DSNUTILB load 148
with FORMAT BMCUNLOAD option 137
with identity columns 113
with SQLAPPLY option 148
constraints, informational 146, 263, 712
constraints, referential
  authorization for checking 50
  checking 146, 263, 712
discards due to 218, 450
ENFORCE option 263, 712
example of checking 612
object status requirements for checking 69, 71
requirements for checking 147
restart considerations 467
with DSNUTILB load 146
See also referential integrity
constraints, referential, authorization for checking 51
CONTINUE keyword
  of MGEXTENT option 712
  of PRELOAD option 208
  of SHORTMEMORY option 239, 712
CONTINUEIF option 128, 243
CONTROL authority 51
Control-M/Tape product 231
controlling access
  in LOADPLUS 51
CONV keyword of DISCARDS option 221
conventions
  rules for specifying command options 163
conversions
allowable 358
character data examples 376
compared to IBM DB2 LOAD 33
data type, allowable 358
date data examples 376
decimal to character 346, 379
discards due to errors 221
example 520
exit routines for 355
integer to character 344, 345, 379
numeric data examples 376
order of 383
time data examples 376
timestamp data examples 376
copies, image 294
command options 294
created in COMBINED phase 695
created in LOAD phase 693
data sets for 436
inline 294, 499, 712
large number of partitions, considerations 687
memory considerations 687
partition-by-growth table spaces, considerations 436
performance considerations 105, 676
subset of partitions in single copy 712
with LOB data 121, 436
with SQLAPPLY load 96
with XML data 115, 436
See also copy data sets
COPY (COPY pending) status
compared to IBM DB2 LOAD 33
COPYPEND option 270, 294, 712
resetting 74
with LPL or WEPR status 69
copy command options 294
copy considerations following a recovery 160
copy data sets
:allocating dynamically 436
allocating 436
calculating space 436
controlling buffer usage 656
description 45, 436
dynamic allocation
assigning partitions 299, 712
DDTYPE option 397, 762
expiration date 420, 762
GDG names 80, 414, 762
how LOADPLUS determines size 436
limiting GDG versions 422, 762
partition-level 299, 712
retention period 421, 762
for LOB data 121, 436
for partition-by-growth table spaces 436
for XML data 115, 436
overriding default ddnames
command options 300, 302
general description 436
installation options 712
partition-level 299, 712
registering 294, 436
space allocation 436
specifying and using 436
subset of partitions in single data set 712
See also copy data sets, dynamic allocation
copy data sets, dynamic allocation of
naming 81
See also dynamic allocation of data sets
COPY option 97, 294
COPY pending (COPY) status
compared to IBM DB2 LOAD 33
COPYPEND option 270, 294, 712
resetting 74
with LPL or WEPR status 69
copy real-time statistics 152
copy registration
BMCXCOPY table 811
in SYSIBM.SYSCOPY 436
specifying 294
COPY utility after load jobs 688
COPYDDN option
command 300
examples 507, 551
installation 712
COPYLVL option
INDEX

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

command 299
example 626
installation 712
performance 645
COPYPEND option
command 294
installation 712
NO example 583
YES examples 557, 588, 596
COPYSUBSET option 626, 712
COUNT keyword
of APRETLIM option 712
of APRETRYLIM option 190
CREATE privileges 51
creating additional data sets as needed 645
creator names, specifying with INTO TABLE 306
CSV data
considerations for loading 126
data types 324
example 583
field specifications for 128, 324
how to load 126
null field, specifying 128
required format for loading 127
specifying 199
syntax for loading 199
CSV keyword of FORMAT option 199, 583
CURRENT DATE option 324
current date variable 414, 762
current Julian date variable 414, 762
CURRENT TIME option 324
current time variable 414, 762
CURRENT TIMESTAMP option 324
CYCLE YES defined on identity column 273

D

D2U (DB2 Utilities Common Code) 47, 49, 431
LOADPLUS requirements 49
D37 abends 55
SYSDISC data sets 467
SYSERR data sets 467
DASD MANAGER PLUS
deleting work files, considerations 712
dynamic data set allocation considerations 80
statistics, updating 289
data availability, maximizing 87, 189
data classes, SMS 422
data conversions
allowable 358
case data examples 376
cased compared to IBM DB2 LOAD 33
date data examples 376
decimal to character 346, 379
discards due to errors 221
example 520
exit routines for 355
integer to character 344, 345, 379
numeric data examples 376
time data examples 376
timestamp data examples 376
data formats, default 324, 379
data from UNLOAD PLUS
considerations 132
example 588
performance benefit from 675
specifying 132, 197
data integrity, checks for 28, 86
data migration, high-speed
example 588
performance considerations 675
specifying 132, 197
using 132
DATA option
keywords
DATA option 183
data set allocation size
limiting 402
data set authorization 50
data set names, reserved 436
data set type, specifying for dynamic allocation 404, 762
data sets
LOADPLUS
spanned 201
variable-block spanned (VBS) 201
data sets, DB2
allocation 45
buffer usage 656
extending 712
preformatting unused portion 271
redefining 246, 712
data sets, LOADPLUS
allocating 432
allocating dynamically 78, 393
APTDUMP 435
ASUSRPT 289, 436
BMCFORCE 436
calculating number 662
cleaning up unused 645
copy 436
creating additional 645
DD statements 432
deleting 82, 250, 712
description of 45, 432
discard (SYSDISC) 450
disposition (DISP parameter)
copy data sets 436
effect on DELETEFILES option 250
effect on DELFILES option 712
restart considerations 467
SORTOUT data sets 467
SYSDISC data sets 450
SYSERR data sets 467
SYSUT1 data sets 467
temporary data sets 85
dummy
copy data sets 294
file size 231
handling 712
SORTOUT restriction 443
SYSERR restriction 453
SYSREC data sets 456
SYSUT1 restriction 459
with INDSN option 211
dump 435
dynamic allocation 78, 393
error (SYSERR) 453
estimating size with ANALYZE option 258, 465, 653
extended 446
extended sequential 762
FASTSWITCH process 102
index (SYSUT1) 459
input (SYSREC) 456
large format 762
LBI 656, 712
load (SORTOUT) 443
mapping (SYSMAP) 454
multiple 670
overview 45
prefixes, specifying 81
redefining 258
rename process 100
reserved names 436
restart considerations 467
SMS-managed 446
SNAP dump 435
SORTOUT 443
SORTWK 446
specifying DD statements 432
staging 98
switching process 102
SYSDISC 450
SYSERR 453
SYSMAP 454
SYSREC 456
SYSUT1 459
uninitialized SYSREC 456
See also individual data set types (for example, SORTOUT data sets)
data sets, REORG PLUS
extended sequential
allocating 404
large format 404
data sharing
group attachment name 427
restarting 467
data translation
group attachment name 427
example 607
processing 383
specifying options 203, 648
See also Unicode
data types
data-partitioned secondary indexes (DPSIs) 225, 661
data-sorting indexes
definition 225
duplicates in 244
ordering by 225
Database Administration for DB2 solution 26
database name variable 414, 762
database status 69
DATAACLAS option 422, 762
date
  conversions, sample 376
  data type 349
  exit routines 668
  expiration 420, 762
  formats 356
  value in column 324
  variables for DSN pattern 414, 762
DATE data type 349
date exit routines
  ASCII
date exit routine 152
time exit routine 152
  EBCDIC
date exit routine 152
time exit routine 152
  security exit 152
time exit routines 152
  Unicode
time exit routine 152
DATE SQL function 142
DAY SQL function 142
DB2
  authorization needed to execute LOADPLUS 50
catalog 102, 110
  check constraints 146, 263, 712
  compression dictionary 140
data sets. See data sets, DB2
dsnutilb program 57
  exits, user 668
LOAD utility 33, 49, 57
  log 33, 94
  optimizer, influencing 291
  real-time statistics 152
  referential constraints 146, 263, 712
  reorg utility input 199
  security exit 51
  system setup 49
  table check constraints 146, 263, 712
  template control statements 60
  user exits 134, 152, 668
  versions supported 49
  versions, when different on data sharing group 427
DB2 Solution Common Code
  LOADPLUS requirements 49
DB2 Solution Common Code (SCC) 47, 49, 431
DB2 Utilities Common Code (D2U) 47, 49, 431

LOADPLUS requirements 49
DBCLOB data type 354
DBCLOBf keyword of character data types 338, 340, 712
DBCS (double-byte character set) 306
DBETE (DBET error) status 69, 71
DD DUMMY 289
DD statements
  $sortparm 465
  aptdump 435
  asusrprt 436
  BatchPipes 139
  Bmcforce 436
  copy data sets 436
  sortout 443
  sortwk 446
  specifying 432
  steplib 431
  sysdisc 450
  sysserr 453
  sysin 454
  sysmap 454
  sysprint 455
  sysrec 456
  sysut1 459
  using with dynamic allocation
    data sets other than sysrec 400, 762
    sysrec 211
  utprint 464
ddname variable 414, 762
ddnames, overriding
  local copy data sets 300, 712
  remote copy data sets 302, 712
  sortout 213, 712
  syssdisc 217, 712
  sysserr 216, 712
  sysidcin 249, 712
  sysmap 284, 712
  sysrec 212, 712
  sysut1 215, 712
DDTYPE option
  command 397
  examples 562, 569
  installation 762
deactivating dynamic data set allocation 398, 762
DECFLOAT data type 284, 354
DECFLOAT_ROUNDMODE option 284
DECIMAL data type 346, 379
decimal floating-point data type 284, 354
decimal point
  as comma 316
  when specifying scale 379
default
  data formats 324, 379
  data types 324
  lengths of input values 358
  values, loading 324
DEFAULTIF option
  description 324
  examples 520, 527
  how RULES option affects 821
  performance considerations 665
defaults, user-defined 324
DEFINE NO objects 467
DEFINE NO, objects created with
  authorization required 51
DEFINE NO, objects defined with
  check constraints 148, 467
  compared to IBM DB2 LOAD utility 33
  in an ERP environment 676
  LOB spaces 122
  object status 69, 71
  partitioned table spaces 676
  performance consideration 676
  restart considerations 467
  status when using SKIPIX option 206
  VALIDPROCs 467
XML spaces 115
DEFINE NO, XML indexes that are defined with 57
DELETE keyword
  of ORIGDISP option 712
  of ORIGINALDISP option 252
  of TAPEDISP option 712
DELETE, data set disposition 250
  work files 712
DELETEFILES option
  description 250
  examples with SYSDISC YES option 569, 588, 602
  with dynamic allocation 82
deleting discard file
  command option 250
  examples 569, 588, 602
  installation option 712
deleting duplicates 244
deleting original data sets after renaming or
  switching staging data sets 252, 712
deleting work files
CHANGE MANAGER considerations 712
command option 250
DASD MANAGER considerations 712
dynamically allocated files 82
installation option 712
performance considerations 645
worklist environment considerations 712
DELFILES option 645, 712
delimited data. See CSV data
delimited date/time formats 356
delimited tokens 163
delimiting characters, for CSV format 127, 199
device types for dynamic allocation, specifying 234, 712
devices, specifying number of 762
dictionaries, compression
  BMCDICT table 141, 792
  building or keeping 187, 306, 712
  how LOADPLUS uses 140
  performance considerations 645
DISABLED keyword of ZIIP option 268, 712
disabling zIIP processing 268, 712
discard data sets (SYSDISC)
  allocating 450
  allocating dynamically 397, 762
  CHANGE MANAGER considerations 450
  default ddname, overriding 217, 712
deleting 250, 712
description 45
  estimating size 258
  expiration date 420, 762
  GDG names 80, 414, 762
  limiting GDG versions 422, 762
  overriding default ddname 217, 712
  restart considerations 467
  retention period 421, 762
  SMS extended sequential data sets 450
  specifying contents 218
  worklist environment considerations 450
DISCARD keyword
  of DDTYPE option 397, 762
  of IDERROR option 275
discard violations, definition 218
DISCARDNN option 217, 507, 551
discard duplicates 244
DISCARDLIMRC option 712
DISCARDRC option 712
discards
designating limit 218
designating return code
  command options 208, 223
determining final 158
  hierarchy 158
installation options 712
file size estimation consideration 231
ignoring 221, 533, 557
types 218, 221, 450
DISCARDS option
  description 218
  examples 533, 557
  performance considerations 658
  with ENUMROWS option 234
DISCDDN option 712
DISPLAY privileges 50, 51
displaying status of BMC utilities 791
disposition (DISP parameter)
  with DELFILES option 712
disposition of data sets (DISP parameter)
  copy data sets 436
  effect on DELETEFILES option 250
  effect on DELFILES option 712
  restart considerations 467
SORTOUT data sets 467
SYSDISC data sets 450
SYSERR data sets 467
SYSREC data set 59
SYSUT1 data sets 467
temporary data sets 85
disposition of data sets (TAPEDISP option) 712
disposition of staging data sets 252, 712
document ID columns 115
See also identity columns
document ID indexes 206
documentation information 13
double-byte character set (DBCS) 306
drain
  canceling threads for 279, 712
  description of processing 77
  number of retry attempts 712
  read claimers, canceling 279, 712
  specifying the number of retry attempts 279
  specifying the time between retry attempts 279
  specifying the timeout value 277, 712
  time between retry attempts 712
  write claimers, canceling 279, 712
DRAIN_WAIT option 277
DRNDELAY option 712
DRNRETRY option 712

DRNWAIT option 712
DSECT, assembler parameter block, in user exit 829
DSN pattern
  concatenation 414
  description 414
  examples 562, 569
  specifying 414, 762
  user-defined variables for 827
DSNHDECP
  implicit time zone value 273
  printing values 427
  settings for changing decimal points to comma 316
  SSID from 427
DSNPAT option
  command 414
  examples 562, 569
  installation 762
DSNTYPE option 404, 762
DSNTYPE=LARGE 456
DSNUEXIT option 395, 712
DSNUTILB load

Index 875
check constraints 148
command options, incompatible 62
command options, translated 62
compression 306
considerations 57
data set name pattern, symbolic variables 414, 762
diagram of phases 38
dynamic data set allocation 60, 78
example 640
field specification considerations 324
how to run 58, 283
incompatible command options 62
INTO statement considerations 306
multi-table table spaces 59
object status 68
options, incompatible 62
options, translated 62
processing phases 38
recovery 478
referential constraints 146
requirements for 58
restarting 467
restrictions 59
software requirements 58
statistics 152, 291
symbolic variables for data set name pattern 414, 762
threads used 56
user-defined variables 414
when invoked 57
DSNUTILB option 283
invoking DSNUTILB 712
DSNUTILB phase 38
DSNZPARMs
   FASTSWITCH value 62, 253, 712
   utility timeout value 277
dummy data set
   file size 231
   handling 712
   SORTOUT restriction 443
   SYSErr restriction 453
   SYSREC data sets 456
   SYSUT1 restriction 459
   with INDSN option 211
dump
   SNAP 95, 435
   specifying DD statement 435
   system 712
DUPKEY keyword of DISCARDS option 221
duplicates
deleting 244
   in data-sorting indexes 244
   in identity columns 273
   in non-data-sorting indexes 244
   in SYSIBM.SYSCOPY 294
   restart considerations 467
   SKIPPIX SIX option considerations 206
   statistics not reflected 289, 291
   when checked and discarded 244
   with DISCARDS IGNORE option 221, 658
   with DISCARDS REPORT option 222
   with INDEX UPDATE option 185
   with SQLAPPLY load 96
   with UNIQUECHECK option 244
DYNALLOC option, RETRY parameter 783
DYNALOC installation option 235, 446, 783
dynamic allocation of data sets
activating or deactivating
data sets other than SYSREC 79, 398, 762
SORTWK 235
SYSREC 211
ACTIVE option 398, 762
ANALYZE option 258, 465
AUTOENUMROWS option 712
automating input file size estimation 231, 712
AVGVOLSP option 762
benefits of 78
CATALOG MANAGER considerations 80
CHANGE MANAGER considerations 80, 398
command options
ACTIVE option 398
ANALYZE option 258
DDTYPE option 397
DSNPAT option 414
DSNUEXIT option 395
ENUMROWS option 229
EXPDT option 420
GDGLIMIT option 422
IFALLOC option 400
INDSN option 211
MAXEXTSZ option 402
MAXTAPE option 394
RETPD option 421
SIZEPCT option 410
considerations 78
copy data sets
assigning partitions 299, 712
for partition-by-growth table spaces 108, 436
GDG names 80, 414
how LOADPLUS determines size 436
installation options 712, 762
overriding default local ddnames 300, 712
overriding default remote ddnames 302, 712
specifying in command option 397
COPYDDN option 300, 712
COPYLVL option 299, 712
DASD MANAGER PLUS considerations 80
data set type, specifying 404, 762
DB2 data sets 45
DDTYPE option 397, 762
deactivating 398, 762
deleting data sets 82, 250, 712
description 78, 393
device types for 234, 712
devices, specifying number of 762
discard files (SYSDISC)
command option 397
deleting 250, 712
GDG names 80
installation option 762
DSNPAT option 414, 762
DSNUEXIT option 395
DSNUTILB load 60
enabling
data sets other than SYSREC 79, 398, 762
SORTWK 235
SYSREC 211
ENUMROWS option 229
error files (SYSERR) 80, 397, 762
examples 562, 569, 588
EXPDT option 420, 762
expiration date 420, 762
extended sequential 404, 762
GDG names
copy, discard, error, and mapping data sets 80, 414, 762
SYSREC 211
GDGEMPTY option 762
GDGLIMIT option 422, 762
GDGSCRATCH option 762
IFALLOC option 400, 762
INDDN option 212, 712
index work files (SYSUT1)
command option 397
installation option 762
overriding default ddnames 215, 712
INDSN option 211
input data sets (SYSREC) 211
installation options 760
issues 78
large format 404, 762
limiting GDG versions 422, 762
limiting size 402, 762
load data files (SORTOUT)
command option 397
installation option 762
overriding default ddname 213, 712
LOADDN option 213
LOB data considerations 398, 436, 762
mapping files 80, 414
mapping files (SYSMAP) 397, 762
MAXEXTSZ option 402, 762
MAXTAPE option
merging with DD statements
naming data sets
  DSNPAT option
  DSNUEXIT option
  SYSREC data sets
  user exits
overview
pattern for names
percentages of allocation size
performance
prefix (data set) considerations
RCVYDDN option
RECOVERYDDN option
restart considerations
retention period
RETPD option
SIZEPCT option
SMS classes
SMS option
SMS-managed
SMSUNIT option
sort work files (SORTWK)
space option
tape devices
THRESHLD option
threshold for secondary values
unit count
unit names
UNIT option
UNITCNT option
units, specifying number of
user exits
using
VOLCNT option
volume count
volumes
  specifying average space on each
  specifying maximum number
WORKDDN option
worklist environment considerations
XML data considerations
dynamic allocation of tape units
DYNAMNBR parameter

E

EBCDIC

constants
date exit routine
option
time exit routine

EDITPROCs
  how LOADPLUS invokes
  performance benefits
  specifying state for
  using
  with ANALYZE ONLY option
  with FORMAT BMCUNLOAD option
elapsed time
  effect of ANALYZE option on
  effect of dynamic allocation on
  effect of ORDER option on
  reducing by using FORMAT BMCUNLOAD
  reducing with multitasking
  reducing with single-phase load
electronic documentation
empty partition
empty table space
ENABLED keyword of ZIIP option
enabling dynamic allocation
  data sets other than SYSREC
  SORTWK data sets
  SYSREC data sets
enabling zIIP processing
ENCLOSEDBY keyword of FORMAT CSV option
enclosure characters, for CSV data
encrypted data
END keyword of LONGNAMETRUNC option
end, values for POSITION option
ENFORCE keyword
  of CHEKPEND option
  of COPYPEND option
ENFORCE option
  authorization required
  command
  effect on status setting
  example
  incompatibility with SQLAPPLY option
  restart considerations
enforcing constraints
  check
  informational
  referential
  enqueues, specifying
ENUMROWS option
   description 229
   examples 569, 588, 619
   performance considerations 648, 659
   restart considerations 467
   with DISCARDS option 234
ERP environments, recommendations 676
ERRDDN option
   command 216
   examples 507, 551
   installation 712
error bytes 218, 383, 450
error data sets (SYSERR)
   allocating 453
   allocating dynamically 397, 762
   deleting 250, 712
   description 45
   estimating size 258
   expiration date 420, 762
   GDG names 80, 414, 762
   inadequate space 467
   limiting GDG versions 422, 762
   overriding default ddname 216, 712
   retention period 421, 762
   specifying DD statements 453
error handling
   controlling discards 218
   FIELDPROCs 450
   identity columns 275
ERROR keyword
   of DDTYPE option 397, 762
   of DEFAULTIF option 324
   of NULLIF option 324
error retry limit, with SQLAPPLY load 190, 712
ERRORBYTE field 218, 383, 450
estimating data set sizes, with ANALYZE option 258, 465, 653
estimating input file size
   automating 231, 712
   LOB data, specifying at the field level
      BLOB field 352
      BLOB file 338, 340
      CLOB field 353
      DBCLOB field 354
   LOB data, specifying globally 287, 712
   performance considerations 648
   specifying estimate 229
   XML data 288, 355, 712
example LOADPLUS jobs 483
example user exits 826
EXCHANGE option 256
   exchange process for clone tables 38, 256
EXCLDUMP option 712
   excluding input fields 128, 304, 324
EXEC statement 426
EXECUTE privileges 50
   High-speed Apply 51
   LOADPLUS 51
   executing LOADPLUS jobs 466
   execution phases of LOADPLUS
      compared to IBM DB2 LOAD 33
      diagrams 38
      primary functions 38
      tuning considerations 689
EXIT data type 355
exit routines
   data type conversion 355
   DB2 152
   how LOADPLUS invokes 152
   invoking MODESET for 668
   user-written
      data type 355
      examples 825
      program language for 395, 712
      return codes 827
      specifying 395, 712
      variables for DSN patterns 414, 762, 827
      with FORMAT BMCUNLOAD option 134
exits
   security 51
expansion errors 383
EXPDT option 420, 762
expiration date 420, 762
expression, indexes created on 57, 142, 712
EXTENDED BUFFER MANAGER. See XBM
extended partitioned data set. See PDSE
extended RBA and LRSN values 162
extended sequential data set
   sallocating 404
   extended sequential data sets
      allocating 762
      SORTOUT data sets 443
      SORTWK data sets 446
      SYSDISC data sets 450
      SYSERR data sets 453
      SYSUT1 data sets 459
   extending data sets 712
   extents
      secondary 712
      specifying maximum size 402, 762
EXTERNAL keyword
  of DATE data type 349
  of DECIMAL data type 346
  of DECIMAL data type 346
  of DECIMAL data type 346
  of DECIMAL data type 346
  of TIME data type 351
  of TIMESTAMP data type 350
  of TIMESTAMP WITH TIME ZONE data type 350
EXTPREF keyword of DSNTYPE option 404, 762
EXTREQ keyword of DSNTYPE option 404, 762

FIELDPROCs
  errors 218, 450
  how LOADPLUS invokes 152, 668
  statistics when using 289
  using 431
  when applied 324
  with FORMAT BMCUNLOAD option 134
FIELDSPEC keyword of DISCARDS option 221
FILECHK option 712
FILEVAR option 712
final return code 158
fixes applied, generating a report for 427
FLOAT data type 347
floating point formats 347, 354
FORCE option 279, 712
FORCE_AT option
  command, of FORCE option 279
  installation 712
FORCE_DELAY option 279
FORCE_RPT option
  command, of FORCE option 279
  installation 712
FORCEROLLUP Common Statistics component option 291
foreign keys, discards 218, 221
format
  ASCII 204
  CSV 126, 199
  EBCDIC 203
  from DB2 REORG 199
  from REORG PLUS 197
  input 196
  internal, from UNLOAD PLUS 132, 197
  reordered row 143
  specifying 196
  SYSREC data sets 456
  Unicode 123, 204
FORMAT BMCLOAD option (in UNLOAD PLUS) 588
FORMAT option

FASTSWITCH option 253, 619, 712
FASTSWITCH process
  authorization when using 50
  description 102
  naming conventions 102
  object status following 72
  option 253, 712
  performance considerations 678
  recovering 478
  restart considerations 467
  specifying when to begin 255
  with ORIGDISP 712
  with ORIGINALDISP 252
FASTSWITCH value in DSNZPARMs 62, 253, 712
features of LOADPLUS 25
field byte lengths 358
field name option 324
field specifications
  default 324
  discards due to errors 221
  for CSV data 128
  for DSNUTILB load 324
  for LOB data 201
  for XML data 201
  not specifying, considerations 324
  options 324
  when spanned 201
  with identity columns 113
field types 358

FAIL keyword
  of FILECHK option 712
  of IDERROR option 275, 712
  of SHORTMEMORY option 239, 712
FILECHK option 712
FILEVAR option 712
file size estimation 231, 712
fast switches 50
fastswitch description 102
fastswitch naming conventions 102
fastswitch object status following 72
fastswitch option 253
fastswitch process authorization when using 50
fastswitch description 102
fastswitch naming conventions 102
fastswitch object status following 72
fastswitch option 253
fastswitch process
  authorization when using 50
  description 102
  naming conventions 102
  object status following 72
  option 253, 712
  performance considerations 678
  recovering 478
  restart considerations 467
  specifying when to begin 255
  with ORIGDISP 712
  with ORIGINALDISP 252
fastswitch value in DSNZPARMs 62, 253, 712
features of LOADPLUS 25
field byte lengths 358
field name option 324
field specifications
  default 324
  discards due to errors 221
  for CSV data 128
  for DSNUTILB load 324
  for LOB data 201
  for XML data 201
  not specifying, considerations 324
  options 324
  when spanned 201
  with identity columns 113
field types 358
BMC keyword 197
BMCUNLOAD keyword
  considerations 132
discards 218, 450
example 588
  identity column considerations 113
  performance considerations 675
  syntax 197
  using 132
CSV keyword 126, 199, 583
  syntax 196
UNLOAD keyword 199
formats
  date 356
  floating point 347, 354
time 356
timestamp 356
formats, floating point 284
fractional portion of decimal value, removing 346
FREE data set allocation parameter 450, 456
FREE keyword of IFALLOC option 400, 762
free pages, when running SQLAPPLY 95
free space
  handling specifications for XML objects 33
  of nonleaf index pages 257
  with SQLAPPLY load 95
FREEPAGE specification, with XML objects 33
FRGNKEY keyword of DISCARDS option 221
FULL keyword of COPYLVL option 299, 712
functions, SQL, indexes created on expressions that contain 142

G

GDGEMPTY option 762
GDGLIMIT option
  command 422
  example 562
  installation 762
GDGs (generation data groups)
  deleting 712
GDGs (generation data groups)
  creating data set names 80, 414, 762
  deleting 250
  limiting number of versions 422
  specifying for input data sets 211
  uncataloging 762
  with partitions 299
GDGSCRATCH option 762
GENERATED ALWAYS columns in temporal tables 306
GENERATED ALWAYS identity columns, overriding 276
GENERATED ALWAYS row change timestamp columns 324
GENERATED ALWAYS ROWID fields 352
  generating identity column values 111
  generating report of applied fixes 427
  generation data groups (GDG). See GDGs
  GRAPHIC data type 343
GRECP (group RECOVER pending) status 71
group name variable 414, 762
group RECOVER pending (GRECP) status 71

H

hardware compression
  BMCDICT table 792
  compared to IBM DB2 LOAD 33
dictionary, building or keeping
  command options 187, 306
  how LOADPLUS operates 140
  installation option 712
  how LOADPLUS uses and performs compression 140
  keeping a dictionary, options 187, 306, 712
  performance considerations 645
hash-organized tables 57, 712
HASHAX option 712
hexadecimal constants 316, 324, 596
HFS, as input 115, 120, 456
hierarchical file system. See HFS
high CPU usage 648, 678
High Speed Utilities (for DB2) solution 26
High-speed Apply Engine
agents 190, 712
authorizations 51
binding packages and plans 190, 712
buffers 679
collection ID 190, 712
commit frequency 190, 712
DB2 authorizations 50
description 47
dump, DD for 435
every 569
options to control 190, 712
owner ID 190, 712
retrying requests 190, 712
SNAP dump, DD for 435
threads 56
using with SQLAPPLY load 93, 190
version required 49
High-speed Apply Engine authorizations 50
high-speed data migration
every 588
performance considerations 675
specifying 197
using 132
high-used RBA
determining size of dynamically allocated files 436
resetting 246, 712
HISTORY Common Statistics component option 291
HISTORY installation option
BMCHIST table 794
HISTRETN installation option
BMCHIST table 796

I
I/O buffers 672, 712
I/O processing, maximizing 672
IBM DB2 LOAD utility 33
IBUFFS option 656, 712
ICOPY (informational COPY pending) status 69, 74
IDCACHE option 273, 712
IDCAMS input data sets (SYSIDCIN)
description 45
overriding default dname 712
IDCAMS, specifying commands for data set redefinition 462
IDCDDDN option 249, 712
identity columns
authorization required 50
cache size for values 111, 273
check constraints 113
compared to IBM DB2 LOAD 33
defaults 113
duplicates 273
error handling 113, 275
field specifications 113
generating values 111
how LOADPLUS updates
MAXASSIGNEDVAL field 111
loading from an input file 111, 276
MAXASSIGNEDVAL field, updating 111, 274, 712
operational considerations 110
order of input values 113, 456
overriding GENERATED ALWAYS columns 276
performance considerations 111, 676
range errors 113, 221, 275
referential constraints 148
resetting the cache 111
table check constraints 113
updating MAXASSIGNEDVAL field 111, 274, 712
validation of values 113
with DSNUTILB load 110
with FORMAT BMCUNLOAD option 113, 137
with SQLAPPLY load 111, 113
identity columns, authorization required 51
IDENTITYOVERRIDE option 276
IDERROR
command option 275
installation option 712
keyword of DISCARDS option 221
IDs
authorization 190, 712
collection 190, 712
RACF 50
subsystem 414, 427, 762
user 414, 427, 762
utility 414, 427, 762
XBM subsystem 269, 712
IEEE Binary Floating Point format (BFP) 347
IFALLOC option
command 400
example 562
installation 762
IFDISCARDLIMIT option 224
IFDISCARDS option 223, 533
IFZEROROWS option 225
IGNORE keyword of DISCARDS option 221, 533, 557
IGNOREFIELDS option of IBM DB2 LOAD utility 304
ignoring discards 221, 533, 557
ignoring indexes 206, 596
image copies. See copies, image
image copy data sets. See copy data sets
image copy data sets, dynamic allocation of
naming 81
implicit time zone 273
IMPLICIT_TIMEZONE DSNHDECP value 273
IMPLICIT_TZ option 273
inadequate space errors 467
incompatible command options
  with DSNUTILB load 62
  with LOB load 121
  with SQLAPPLY option 97
  with XML load 115
INDDN option
  command 212
  examples 507, 551, 626
  installation 712
  with INDSN option 211
index data sets (SYSUT1)
  allocating 459
  allocating dynamically 397, 762
  CHANGE MANAGER considerations 459
  controlling buffer usage 656
  default ddname, overriding 215, 712
  deleting 250, 712
  description 45
  estimating size 258
  LOB requirements 459
  multiple 459, 670
  overriding default ddnames 215, 712
  performance considerations 656, 670
  single-phase load use 459
  SMS extended sequential data sets 459
  specifying DD statements 459, 670
  tuning considerations 656, 670
  worklist environment considerations 459
  XML requirements 459
INDEX option
  example 551
    performance considerations 648, 664
    syntax description 185
    with SQLAPPLY option 97
index pages, controlling free space in nonleaf 257
index processing
  compared to IBM DB2 LOAD 33
  during LOAD phase 693
  during PRELOAD phase 691, 695, 698
  restart considerations 467
  skipping 206, 596
index space, status 68
index statistics 152
indexes
  assigning to index data sets 459
  authorization 51
  auxiliary 206
  building 185, 693
  clustering 225, 670
  created on expression 57, 142, 712
  data-partitioned secondary (DPSIs) 225, 661
  data-sorting 225, 244
  document ID 206
  free space, controlling 257
  ignoring 206, 596
  in LPL or WEPR status 69
  LOB 206
  multiple 670
  node ID 206
  non-data-sorting, duplicates in 244
  non-key columns in 712
  non-unique, ignoring 206, 596
  nonparticipating 206, 596
  nonpartitioned, ignoring 206, 596
  null keys 57, 712
  partitioned partitioning 459, 670
  partitioning, ordering by 225
  primary of the parent table 148, 161
  random 57, 640, 712
  redefining 246
  referential constraints, using to check 148, 161
  reorganizing while loading 648
  row ID 206
  sampling 258
  scanning 258
  skipping 206, 596
  statistics 152
  status
    primary of the parent table 69, 71
    set when SKIPIX specified 74, 206
  tuning 664
  updating 185
  versioned 134
  XML 57, 206
INSDN option
  description 211
  examples 499, 588
  with INDDN option 212
INFORI
  installation option 712
  keyword of ENFORCE CONSTRAINTS option 263
  keyword of ENFORCE RI option 263
informational COPY pending (ICOPY) status 69, 74
informational referential constraints
  command option 263
  enforcing 146
  installation option 712
  preventing CHKP 151
  restoring integrity 151
See also referential constraints
inline image copies
  created in COMBINED phase 695
  created in LOAD phase 693
  example 499
  specifying 294, 712
inline LOB columns 142
inline LOB data 712
INLINE option 294, 499
INLINECP option 712
INLOB option 712
input data
  batch pipe 139, 218, 450
  compressed 231
  CSV 126, 199, 583
  DB2 REORG utility 199
  default lengths of values 358
  delimited 126, 199, 583
  identity columns 110
  inline LOB 712
  LOB 119
  ordering 225
  pipe data 139, 218, 450
  REORG PLUS archive data set 197
  sorting 225
  specifying format 196
  UNLOAD PLUS 132, 197, 588
  XML 114, 632
input data set 454
input data sets
  spanned-record format 201
  VBS 201
input data sets (SYSREC)
allocation 456
  allocating dynamically 211
  batch pipe data 139, 218, 450
  calculating file size 229, 712
  calculating number 456, 662
  concatenating 675
  considerations 456, 670
  controlling buffer usage 656
  default ddname, overriding 212, 712
  description 45
  estimating file size
    AUTOENUMROWS option 712
    AVGSIZE option 352–355
    ENUMROWS option 229
    LOBAVGPT option 287, 712
    XMLAVGSIZE option 288, 712
  format 456
  GDG 211
  INSDN option 211
  multiple
    enabling multitasking 670
    performance considerations 675
    specifying 456
    specifying prefix for 212, 712
  overriding default ddname 212, 712
  performance considerations 675
  pipe data 139, 218, 450
  referencing files in 456
  restart considerations 467
  restrictions 456
  specifying DD statements 456
  specifying format 196
  specifying referenced files in 456
  with DSNUTILB load 59
INSERT privileges 50, 51
inserting multiple rows in a single insert action,
  with SQLAPPLY load 190, 712
INSTALL SYSADM authority 51, 712
installation options
  $AMUDYNA macro 760
  BMCHIST 794
  BMCSORT 703
  definitions and default values 703
  dynamic data set allocation 760
  HISTORY 794
  HISTRETN 796
  multiple modules 427, 703
  setting for optimal performance 645
  specifying a module 427
Installation System 703
installing LOADPLUS 703
INSTALLSYSADM keyword of UPDMAXA_AUTHID option 712
INTEGER data type 344, 379
integrity
data, checking 28, 85
referential. See referential integrity
restoring referential and table check 151
table check. See table check constraints
INTO TABLE option
description 304
multiple partitions 306
multiple statements
discard implications 218
example 527
with ENUMROWS AUTO 233
with UNIQUEINTO option 305
performance considerations 660
preformatting data sets 306
replacing tables 306
invoking DSNUTILB 57, 283
See also DSNUTILB load
invoking LOADPLUS 467
IRLMWAIT, used with drain timeout 277
IXINCLCOL option 712
IXONEX option 712
IXRANDOM option 712

J
JCL for LOADPLUS examples, where to find 483
JES3 considerations 45
JES3 limitation 464
JOB name variable 414, 762
JOB statement 425
jobs, LOADPLUS
building 425
canceling 477
configuration 703
examples 483
executing 466
restarting 467
running 466
terminating 477
Julian date variables 414, 762

K
KEEPDICTIONARY option
ACTIVE option 398, 762
ALL keyword
  of DISCARDS option 221
  of FORCE option 279, 712
  of REGISTER option 294
ALLTYPES keyword of SKIPFIELDS option 304
ANALYZE keyword of PRELOAD option 208
ANALYZE option 258
AND keyword of FORMAT CSV option 199
ANYDISCARDS option 208
APCOLLECTION option
  installation 712
  of SQLAPPLY option 190
APCOMMIT option
  installation 712
  of SQLAPPLY option 190
APDOPTS option 712
APMAXAGENTS option of SQLAPPLY option 190
APMULTIROW option
  installation 712
  of SQLAPPLY option 190
APMVXAGNT option 712
APOWNER option
  installation 712
  of SQLAPPLY option 190
APRETLIM option 712
APRETRYLIM option of SQLAPPLY option 190
APRETRYVAL option of SQLAPPLY option 190
APRETV AL option 712
ASCII option 204
ASM keyword of DSNUEXIT option 395, 712
ASSOCIATE option of ORDER YES option 225
AUTO keyword
  of ENUMROWS option 231
  of VOLCNT option 408, 762
AUTOENUMROWS option 712
AVAILPAGETPCT option 237, 712
AVGSIZE keyword
  of BLOB data type 352
  of BLOBF subtype 338, 340
  of CLOB data type 353
  of CLOBF subtype 338, 340
  of DBCLOB data type 354
  of DBCLOBF subtype 338, 340
  of XML data type 355
AVGVOLSP option 409, 762
BASIC keyword
  of DSNTYPE option 404
BASIC keyword of DSNTYPE option 762
BEGINNING keyword of
  LONGNAMETRUNC option 202, 712
BIGINT data type 345
BINARY data type 348
BINARY VARYING data type 348
BINARYXML option
  of CHAR data type 338
  of VARCHAR data type 340
  of XML data type 355
BLOB data type 352
BLOBF keyword of character data types 338, 340
BMC keyword
  of FORMAT option 197
  of NLPCFTFREE option 257
  of RULES option 712
BMCSSTATS option 289
BMCDUNLOAD option 197
BUILD keyword of INDEX option 185
BYCLUSTERKEY keyword of ORDER YES
  ASSOCIATE option 225
BYTABLE keyword of ORDER YES
  ASSOCIATE option 225
C keyword of DSNUEXIT option 395, 712
CBUFFS option 712
CCSID option 204
CENTURY option
  global 272
  installation 712
  of DATE data type 349
  of TIMESTAMP data type 350
CHANGE keyword of SHRLEVEL option 190, 712
CHAR data type 338
CHECK CONSTRAINTS option 263
CHECK keyword of ENFORCE option 712
CHECKPEN D option 267
CHEKPEN D option 712
CHKC keyword of DISCARDS option 221
CLOB data type 353
CLOBF keyword of character data types 338, 340
CLUSTER keyword of UNIQUECHECK option 244
COBOL2 keyword of DSNUEXIT option 395, 712
CONSTRAINTS keyword of ENFORCE option 263, 712
CONTINUE keyword
FORCE_DELAY option of FORCE option 279
FORCE_RPT option
command, of FORCE option 279
installation 712
FORMAT option 196
FREE keyword of IFALLOC option 400, 762
FRGNKEY keyword of DISCARDS option 221
FULL keyword of COPYLV option 299, 712
GDGEMPTY option 762
GDGLIMIT option 422, 762
GDGSCRATCH option 762
GRAPHIC data type 343
HASHAX option 712
IBUFFS option 712
IDCACHE option 273, 712
IDCDN option 249, 712
IDENTITYOVERRIDE option 276
IDERROR
command option 275
installation 712
keyword of DISCARDS option 221
IFALLOC option 400, 762
IFDISCARDLIMIT option 224
IFDISCARDS option 223
IFZEROROWS option 225
IGNORE option 221
IMPLIED_TZ option 273
INDDN option 212, 712
INDEX option 185
INDSN option 211
INFORI
installation 712
keyword of ENFORCE CONSTRAINTS option 263
keyword of ENFORCE RI option 263
INLINE option 294
INLINECP option 712
INLOB option 712
INSTALLSYSADM keyword of
UPDMAXA_AUTHID option 712
INTEGER data type 344
INTO TABLE option 306
IXINCLCOL option 712
IXONEX option 712
IXRANDOM option 712
KEEPDICTIONARY option
global 187, 712
of PART option 306
LARGE keyword of DSNTYPE option 404, 762
LASTRETRY keyword of FORCE_AT option
LOAD command 183
LOAD keyword
of DDTYPE option 397, 762
of PRELOAD option 208, 712
LOADDDN option 213, 712
LOADLOBDATA option 286
LOB option 712
LOBAVGCP option 287, 712
LOCBFCPY keyword of DDTYPE option 397, 762
LOBCLCPY keyword of DDTYPE option 397, 762
LOCPXCPY keyword of DDTYPE option 397, 762
LOG option 270
LONGNAMETRUNC option 202, 712
MAPDDN option 284, 712
MAXEXTSZ option 402, 762
MAXP option 712
MAXSORTMEMORY option 242, 712
MAXSORTS option 236
MAXTAPE option 394, 712
MGEXTENT option 414
MGMTCLAS option 422
MIDDLE keyword of LONGNAMETRUNC option 202, 712
MINSORTMEMORY option 241, 712
MIXED keyword
of CHAR data type 338
of CLOB data type 353
of VARCHAR data type 340
MSGLEVEL option 712
NLPCTFREE option 257
NOCOPYPEND option 270
NONE keyword
of DRAIN_WAIT option 277
of DRNWAIT option 277
of DSNPAT option 414
of DSNUEXIT option 395, 712
of FORCE option 279, 712
of REGISTER option 294
of SHRLEVEL option 190, 712
NOSUBS option 205
NOW keyword
  of EXCHANGE option 256
  of SWITCHTIME option 255
NULLIF option 324
NUSIX keyword of SKIPIX option 206
ONLY keyword of ANALYZE option 258
OPNDB2ID option 712
ORDER option 225
ORIGDISP option 712
ORIGINALDISP option 252
PACKED keyword of DECIMAL data type 346
PARM option of EXIT data type 355
PART
  command option 306
  keyword of COPYLVL option 299, 712
  keyword of DISCARDS option 221
PAUSE keyword
  of ANALYZE option 258
  of EXCHANGE option 256
  of PRELOAD option 208
  of SWITCHTIME option 255
PAUSEDISCARDRC option 712
PERIODOVERRIDE option 306
PLAN option 712
POSITION option 324
PREFORMAT option
  global 271, 712
  of PART option 306
PRELOAD option 208, 712
PRESERVE WHITESPACE keyword
  of BLOBF subtype 338, 340
  of CLOBF subtype 338, 340
  of DBCLOBF subtype 338, 340
  of XML data type 355
PRESORTED keyword of ORDER option 225
PROB keyword of UXSTATE option 712
RCVYDDN option 712
READERS keyword of FORCE option 279, 712
RECOVERYDDN option 302
REDEFINE option 246, 712
REFERENCE keyword of SHRLEVEL option 190, 712
REGISTER option of COPY YES option 294
REMBFCPY keyword of DDTYPE option 397, 762
REMBLCPY keyword of DDTYPE option 397, 762
REMBXCPY keyword of DDTYPE option 397, 762
REMOVEDECIMAL option 346
REMOVESIGN option
  of BIGINT data type 345
  of DECIMAL data type 346
  of INTEGER data type 344
  of SMALLINT data type 344
REMPFCPY keyword of DDTYPE option 397, 762
REMLCPY keyword of DDTYPE option 397, 762
REMPXCPY keyword of DDTYPE option 397, 762
RENAME keyword
  of ORIGDISP option 712
  of ORIGINALDISP option 252
RENMMAX option 208
REPLACE option
  global 187
  of PART option 306
  of TABLE option 306
REPORT keyword
  of BMCSTATS option 289
  of DISCARDS option 222
RESET keyword of MGEXTENT option 712
RESUME option 183
RETCODE option 208
RETPD option 421, 762
RETRY keyword of FORCE_AT option 279, 712
RETRY option 279
RETRY_DELAY option 279
REUSE option 249
RI keyword of ENFORCE option 263, 712
ROUND option
  of BIGINT data type 345
  of DECIMAL data type 346
  of FLOAT data type 347
  of INTEGER data type 344
  of SMALLINT data type 344
ROUND_CEILING keyword of
  DECFLOAT_ROUNDMODE option 284
ROUND_DOWN keyword of DECFLOAT_ROUNDMODE option 284
ROUND_FLOOR keyword of DECFLOAT_ROUNDMODE option 284
ROUND_HALF_DOWN keyword of DECFLOAT_ROUNDMODE option 284
ROUND_HALF_EVEN keyword of DECFLOAT_ROUNDMODE option 284
ROUND_HALF_UP keyword of DECFLOAT_ROUNDMODE option 284
ROUND_UP keyword of DECFLOAT_ROUNDMODE option 284
ROWID data type 352
RULES option 712
SAMPLE keyword of ANALYZE option 258
SCAN keyword of ANALYZE option 258
SDUMP option 712
SHORTMEMORY option 239, 712
SHRLEVEL option 189, 712
SIX keyword of SKIPIX option 206
SIZEPCT option 410, 762
SKIPFIELDS option 304
SKIPIX option 206
SMALLINT data type 344
SMAX option 712
SMCORE option 712
SMS option 401, 762
SMSUNIT option 402, 762
SORTDEVT option 234, 712
SORTKEYS option 207
SORTNUM option 235, 712
SORTWORK keyword of DDTYPE option 397, 762
SPACE option 411
SPANNED keyword of FORMAT option 201
SQL keyword
of DRAIN_WAIT option 277
of DRNWAIT option 712
SQLAPPLY keyword of SHRLEVEL option 190, 712
SQLDELAY option 712
SQLRETRY option 712
STANDARD keyword
of NLPCTFREE option 257
of RULES option 712
START keyword of FORCE_AT option 279, 712
STOP@CMT option 712
STOPDELAY option 712
STOPRETRY option 712
STORCLAS option 422
SUP keyword of UXSTATE option 712
SWITCHTIME option 255
SYSDISC option of DELETEFILES option 250
SYSMAP keyword of DDTYPE option 397, 762
TABLE=obid option 313
TAPEDISP option 712
TEMPRALDATA option 712
TERMINATEDBY option of FORMAT CSV option 199
THRESHLD option 411, 762
TIME data type 351
TIME keyword
of APRETLIM option 712
of APRETRYLIM option 190
TIMESTAMP data type 350
TIMESTAMP WITH TIME ZONE data type 350
TOTALPAGEPCT option 238, 712
TRANSACTIONIDOVERRIDE option 306
TRIM option
of BINARY data type 348
of BINARY VARYING data type 348
of CHAR data type 338
of VARBINARY data type 348
of VARCHAR data type 340
TRUNCATE option
of BINARY data type 348
of BINARY VARYING data type 348
of CHAR data type 338
of VARBINARY data type 348
of VARCHAR data type 340
TSSAMPLEPCT option 293, 712
UNCATLG keyword of TAPEDISP option 712
UNICODE option 204, 712
UNIQUECHECK option 244
UNIQUEINTO option 305
UNIT option 406, 762
UNITCNT option 407, 762
UNLOAD keyword of FORMAT option 185
UPDATE keyword of INDEX option 185
UPDATEDB2STATS option 291
UPDATEMAXA option 274
UPDMAXA option 712
UPDMAXA_AUTHID option 712
USE keyword of IFALLOC option 400, 762
USER keyword of UPDMAXA_AUTHID option 712
UTIL keyword
  of DRAIN_WAIT option 277
  of DRNWAIT option 712
UTILB_COLCCSID option 712
UTILB_NULLIX option 712
UXSTATE option 712
VALPROC keyword of DISCARDS option 221
VALUE option 324
VARBINARY data type 348
VARCHAR data type 340
VARGRAPHIC data type 343
VOLCNT option 408, 762
WARN keyword of FILECHK option 712
WBUFFS option 712
WHEN keyword of DISCARDS option 221
option of INTO TABLE option 313
WORK keyword of DDTYPE option 397, 762
WORKDDN option 215, 712
WORKUNIT option 712
WTOR keyword
  of EXCHANGE option 256
  of SWITCHTIME option 255
XBLKS option 712
XBIMID option 269, 712
XML data type 355
XML option 712
XMLAVGSIZE option 288, 712
ZEROROWRC option 712
ZIIP option 268
ZONED keyword of DECIMAL data type 346
ZPARM keyword of FASTSWITCH option 712

language, for user exits 395, 712
large format sequential data sets 404, 762
LARGE keyword of DSNTYPE option 404, 762
large number of partitions 105, 141, 676
LASTRETRY keyword of FORCE_AT option 279, 712
LBI data sets 656, 712
LBUFFS option 712
LE C sample user exit 854
LE COBOL sample user exit 839
LE_C keyword of DSNUEXIT option 395, 712
LE_COBOL keyword of DSNUEXIT option 395, 712

lengths of fields 358
lengths of input values, default 358
LIMIT value of TEMPLATE control statement 411, 762
limiting number of GDG generations 422, 762
limiting retry attempts
determining stopped status 712
drain attempts 712
High-speed Apply requests 190, 712
SQL statements 712
limiting size of dynamically allocated data sets 402, 762
LLBnnnnn data sets 436
LLPnnnnn data sets 436
LOAD command 183
load data sets (SORTOUT)
multiple 670
load data sets (SORTOUT)
allocating 443
allocating dynamically 397, 762
calculating number 208, 443, 662
controlling buffer usage 656
default ddname, overriding 213, 712
deleting 250, 712
description 45
estimating size 258
multiple 213, 443
optimal number of 670
overriding default ddname 213, 712
performance considerations 656
single-phase load use 443
specifying DD statements 443
tuning considerations 670
LOAD keyword
  of DDTYPE option 397, 762
  of PRELOAD option 208
LOAD phase
  index building process 693
  invoking BMCSORT 693
load process 693
primary functions 38
recovery 478
resource allocation diagram 693
tuning considerations 693
load process in LOAD phase 693
load processing options 182
LOAD REPLACE option, examples 514
LOAD RESUME YES option
ENUMROWS option recommendations 659
examples 507, 602
INDEX option recommendations 648
object status 71
small incremental loads 688
SQLAPPLY option recommendations 648, 678, 688
syntax description 183
when loading LOB data 122
when loading XML data 115
with ANALYZE option 258
with BMCSTATS option 289
with ENUMROWS option 229
with inline image copies 294
with ORDER option 225
with REDEFINE YES option 246
with SHRLEVEL option 190, 712
with UPDATEDB2STATS option 291
LOAD RESUME YES SHRLEVEL CHANGE
SQLAPPLY load

affecting organization of table space 95
availability recommendations 688
check constraints 148
command options, incompatible 97
considerations 93
DB2 logs 94
diagram of phases 38
dump, SNAP 435
duplicate keys 96
example 569
image copies 96
incompatible command options 97
inserting multiple rows in a single insert action 190, 712
invoking BMCSORT 698
multi-data-set objects 78
multi-row insert 190, 712
multiple table loads 95
object status 71
ORDER option 95
performance recommendations 648, 678
processes 698
referential constraints 146
referential integrity 95
resource allocation diagram 698
restarting 96, 467
SNAP dump 435
statistics 97
syntax description 190
table check constraints 148
threads used 56
tuning 698
when loading LOB data 122
when loading XML data 118
LOAD RESUME YES SHRLEVEL CHANGE
SQLAPPLY, authorizations for 51
LOAD utility of IBM
compared to LOADPLUS 33
IGNOREFIELDS option 304
installation requirement 49
invoking DSNUTILB to use 57, 283, 712
STATISTICS option 291
LOAD utility of IBM, installation requirement 49
LOADDN option
command 213
examples 507, 551
installation 712
loading small amounts of incremental data 688
LOADLOBDATA option 286
LOADPLUS
authorizations needed 51
LOADPLUS compared to IBM DB2 LOAD utility 33
LOB data 352
  BMCSYNC table considerations 801
  considerations for loading 120
  copies for 121, 436
  data types 353, 354
  DATE SQL function 142
  DAY SQL function 142
  example 632
  from referenced files 120, 456, 712
  how to load 119
  incompatible options 121
  inline 142, 712
  length parameter, in UPPER or LOWER SQL functions 142
  loading from spanned-record data sets 201
  locale, in UPPER or LOWER SQL functions 142
  LOWER SQL function 142
  MONTH SQL function 142
  nested functions in an expression 142
  specifying estimated size of data
    BLOB field 352
    BLOB file 338, 340
    CLOB file 338, 340, 353
    DBCLOB field 354
    DBCLOB file 338, 340
    globally 287, 712
    UPPER SQL function 142
    YEAR SQL function 142
  LOB option 712
  LOBAVGPCPT option 287, 712
  LOCBFCPY keyword of DDTYPE option 397, 762
  LOCBLCPY keyword of DDTYPE option 397, 762
  LOCBXCPY keyword of DDTYPE option 397, 762
  LOCKROW option 712
  LOCFCPY keyword of DDTYPE option 397, 762
  LOCPLCPY keyword of DDTYPE option 397, 762
  LOCXCPY keyword of DDTYPE option 397, 762
  Log Master, High-speed Apply Engine component of. See High-speed Apply Engine
  LOG option 97, 270
  log range table 797
  log, DB2 33, 270
  LOGGED table spaces 33, 94
  logical page list (LPL) status 69, 71
  logical part REBUILD pending (RBDP) status 71
  long object names
  syntax rules 163
  truncating in messages 202, 712
  LONGNAMETRUNC option 202, 712
  low cardinality keys 648, 678
  LOWER SQL function 142
  LPL (logical page list) status 69, 71
  LRBnnnnn data sets 436
  LRPhnnnn data sets 436
  LRSN, extended 162

M

MAINT parameter of EXEC statement 427
maintaining common utility tables 789
MainView Batch Optimizer 139
MainView SRM StopX37 product 55
management classes, SMS 422
management classes, SMS keywords
  MGMTCLAS option 762
  MAPDDN option 284, 712
mapping data set (SYSMAP)
  allocating dynamically 397, 762
  default ddname, overriding 284
  description 45, 454
  expiration date 420, 762
  GDG names 80, 414, 762
  limiting GDG versions 422, 762
  overriding default ddname 284
  retention period 421, 762
materializing DEFINE NO data sets
  authorization required 51
  check constraints 148
  how LOADPLUS materializes 33
  LOB spaces 122
  performance considerations 676
  restart considerations 467
  XML spaces 115
MAXASSIGNEDVAL field
  generated identity columns based on 111
  updating value 111, 274, 712
MAXEXTSZ option 402, 762
maximizing data availability 87, 678
maximizing I/O processing 672
MAXP option 661, 712
MAXPRIME keyword of TEMPLATE control statement 402, 762
MAXSORTMEMORY option 242, 712
MAXSORTS option
LOADPLUS for DB2 Reference Manual

N

names of common utility tables, determining 791
names of objects, long
syntax rules 163
truncating in messages 712
names of objects, truncating in messages 202
naming data sets
  GDG names 80
  partition-level copy data sets 436
  prefixes for ddnames 81
  staging data sets 100, 102
  user exits 827
  using a pattern 414, 762
ncopy data sets
  overriding default ddnames
  installation options 712
NEW parameter of EXEC statement 427
NEW/RESTART parameter of EXEC statement 427
NEW/RESTART(PHASE) parameter of EXEC statement 427
NLPCTFREE option 93, 257
NOCOPYPEND option 270
node ID indexes 206
NOLIMIT value of MEMLIMIT system parameter 55, 426, 672
non-data-sorting indexes, duplicates in 244
non-key columns, indexes containing 712
non-unique indexes, ignoring 206, 596
NONE keyword
  of DRAIN_WAIT option 277
  of DRNWAIT option 712
  of DSPAT option 414
  of DSNTYPE option 404, 762
  of DSNUEXIT option 395, 712
  of FORCE option 279, 712
  of REGISTER option 294
  of SHRLEVEL option 190, 712
nonleaf index pages, controlling the amount of free space 257
nonparticipating indexes 206, 596
nonpartitioned indexes, ignoring 206, 596
nonpartitioned table spaces
  with copy DD statements 436
  with FASTSWITCH 253, 712
  with SKIPIX option 206
  with TABLE REPLACE 306
NOSUBS option 205
NOT LOGGED table spaces 33, 94
NOW keyword
  of EXCHANGE option 256
  of SWITCHTIME option 255
null
for constants in VALUE option 324
in APCOLLECTION option 190, 712
in APOWNER option 190, 712
in CSV data 128, 324, 583
in FORMAT CSV option 199
in predicates 316
  keys 57, 712
  loading 324
  NULLIF option 324, 821
  translating data to 520
  with DEFAULTIF option 324
NULLFILE data set name 294
NULLIF option
  description 324
  how RULES option affects 821
  performance considerations 665
  with FORMAT CSV option 324
number of CPUs 687
number of devices, specifying 762
number of GDG generations, limiting 422, 762
number of units, specifying 762
number of volumes, specifying maximum 762
numeric data conversions 376, 379
numeric variables 414
NUSIX keyword of SKIPIX option 206

O

OBID (object identifier) 197, 313, 588
Object Administration (for DB2) solution 26
object availability, SHRLEVEL option 189, 688, 712
object identifier (OBID) 197, 313, 588
object names, long
  syntax rules 163
  truncating in messages 202, 712
object status 33, 68
ONLY keyword of ANALYZE option 258
operating system requirements 49
operators
  comparison for predicates 316
  conditional 313
  how RULES option affects 316, 819
OPNDB2ID option 50, 712
options
  statistics 288
  options modules, specifying 427
  options, command

Index 895
basic 182
copy 294
dynamic allocation
    ANALYZE option 258
    ENUMROWS option 229
    for SYSREC data sets 211
    main options 393
field specification 324
INTO TABLE option 304
options, installation 703
order of input values 96, 113
ORDER option
    collating sequence for character data 225
    description 225
    memory usage 687
    performance considerations 661, 675
    with SQLAPPLY load 95, 97
organization of table space, affected by SQLAPPLY 95
ORIGDISP option 712
ORIGINALDISP option 252
out-of-space errors, restart considerations 467
output data set from PRELOAD phase. See
    SORTOUT data sets
overriding default ddnames
    BMCCPY 300, 712
    BMCCPZ 300, 712
    BMCRCY 302, 712
    BMCRCZ 302, 712
    local copy data sets 300, 712
    remote copy data sets 302, 712
SORTOUT 213, 712
SYSDISC 217, 712
SYSERR 216, 712
SYSIDCIN 249, 712
SYSPHYS 284
SYSMAP 284
SYSREC 212, 712
SYSUT1 215, 712
owner ID, High-speed Apply 190, 712

P

packages
    authorization 50
    binding for High-speed Apply request 190, 712
    LOADPLUS 51, 712
PACKED keyword of DECIMAL data type 346
padding, order of 383
page set REBUILD pending (PSRBD) status 71
parallel sort processes
    controlling 665
    optimal number 670
    specifying maximum number 236, 712
parameter block in user exit 829
parameter record in user exit 839
parameter structure in user exit 847, 854
parameters, EXEC statement 427
parent table indexes, authorization for 51
PARM option of EXIT data type 355
PART
    command option 97, 306
    keyword of COPYLVL option 299, 712
    keyword of DISCARDS option 221
partial load
    assigning copy data sets 299, 712
    examples 507, 551
    file size estimation consideration 231
    specifying copy data sets 436, 712
    specifying partitions 306
partition number variable 414, 762
partition-by-growth table spaces
    adding new partitions 108
    BMCSYNC table 110
    compression dictionaries 108
    considerations 108
    copies 108, 294
    copy data sets 436
    DD statements for copies 436
    example 557
    LOB data 57, 122
    restart considerations 110, 427, 467
    SHRLEVEL considerations 109
    SORTOUT data sets 443
    SQLAPPLY considerations 190
    staging data sets 109
partition-level copies
    naming conventions 436
    specifying 299, 712
    specifying prefix for 300, 302
partitioned partitioning indexes 459, 670
partitioned table spaces
assigning image copy data sets 299, 712
copy DD statements 436
DEFINE NO consideration 122, 676
examples 499, 507
High-speed Apply agents 190, 712
large number of partitions considerations 105, 140, 676
memory considerations 687
performance considerations 670, 676, 678
prefix for copy data set ddnames 300
specifying partitions to load 306
subset of partitions in single image copy 712
partitioning key, ordering by 225, 588
partitions
  adding to partition-by-growth table spaces 108
  assigning for copies 299, 712
  copies for 436, 712
  large number 105, 141, 676
  specifying 306
  specifying multiple 306
path name, specifying in input file 115, 120, 456
pattern for dynamically allocated data set names 414, 762, 827
pattern, DSN
  command option 414
  concatenation 414
  installation option 762
  variables for 414, 827
PAUSE keyword
  of ANALYZE option 258
  of EXCHANGE option 256
  of PRELOAD option 208
  of SWITCHTIME option 255
PAUSEDISCARDRC option 712
pausing LOADPLUS jobs
  with ANALYZE option 258, 538
  with EXCHANGE option 256
  with PRELOAD option 208
  with SWITCHTIME option 255
PDS as input
  LOB data 120, 456
  using INDSN 211
  XML data 115, 456
PDSE as input 115, 120, 456
pending statuses
  initial object status 69
  setting or resetting after load 74
percent
ANALYZE option 653
APMULTIROW option 655
AUTOENUMROWS option 648, 659
BMCSORT 665
buffer usage 656
COMBINED phase 663
copy data sets 656
data from UNLOAD PLUS 675
DB2 data sets 656
DEFAULTIF option 665
DEFINE NO data sets 676
DISCARDS option 658
dynamic allocation 659
EDITPROCs 668
ENUMROWS option 659
exit routines 668
FASTSWITCH option 678
FIELDPROCs 668
general considerations 645
I/O processing 672
INDEX option 648
index work files 656, 670
inserting multiple rows in a single insert action, with SQLAPPLY load 655
INTO TABLE option 660
MAXP option 661
messages 679
multiple data sets 670
multitasking 670
NULLIF option 665
ORDER option 661
partitioned table spaces 670
processing phases 689
REPLACE option 664
RESUME option 664
single-phase load 663
sort processing 665
SORTOUT data sets 656, 670
SQLAPPLY load 655, 678
staging data sets 643, 678
SYSREC data sets 656, 670
SYSUT1 data sets 656, 670
UNIQUEINTO option 660
user exits 668
VALIDPROCs 668
virtual storage 672
WHEN option 665
XBLKS option 669
performance related messages 679
PERIOD constraint of a temporal table 306
PERIODOVERRIDE option 306
persistent read only (PRO) status 69, 71
phases of LOADPLUS diagrams 38
primary functions of each 38
tuning considerations 689
pipe input 139, 218, 450
placeholder fields 304
specifying 324
with CSV data 128, 324
PLAN option 712
plans
authorization binding for High-speed Apply requests 190, 712
LOADPLUS 51
specifying 712
POSITION option 324
preallocating HFS 115, 120
preallocating staging data sets 99, 246
predicate option 316
prefix, specifying ddname
general considerations 81
prefixes, data set 432
PREFORMAT option command 271
installation 712
of PART option 306
with SQLAPPLY option 97
preformatting unused portion of DB2 data sets 271, 712
PRELOAD option command 208
examples 487, 507
installation 712
PRELOAD phase
data process 691, 695, 698
index process 691, 695, 698
invoking BMCSORT 691
primary functions 38
recovery in 478
resource allocation diagram 691
tuning considerations 691
PREERVE WHITESPACE keyword of BLOBF subtype 338, 340
of CLOBF subtype 338, 340
of DBCLOBF subtype 338, 340
of XML data type 355
preserving white space
BLOB file 338, 340
DBCLOB file 338, 340
XML input 355
PRESORTED keyword of ORDER option 225
preventing $x37 abends 55
primary indexes of the parent table
referential constraints, using to check 148, 161
requirements when checking referential
constraints 148
status 69, 71
privileges
ALTER 50, 51, 712
CONTROL 50
READ 50
SELECT 50
UPDATE 50
PRO (persistent read only) status 69, 71
PROB keyword of UXSTATE option 712
processing options 182
processing phases
diagrams 38
primary functions of each 38
tuning considerations 689
processor requirements 49
product changes 16
program language, for user exits 395, 712
PSRBD (page set REBUILD pending) status 71
PTFs applied, generating a report for 427
publications, related 13

Q
qualifiers, I and J switching 102

R
RACF
security exit 51
RACF (IBM Resource Access Control Facility)
authorization needed to execute LOADPLUS 50
OPNDB2ID option 712
security exit 50
RACF (IBM Resource Access Control Facility)
authority
LOADPLUS 51
random indexes 57, 640, 712
range errors on identity columns 275, 712
RBA, extended 162
RBA, high-used
determining size of dynamically allocated files 436
resetting 246, 712
RBDP (REBUILD pending) status 71, 596
RBDP* (logical part REBUILD pending) status 71
RCVYDDN installation option 712
read claimers, canceling 279, 712
READ privileges
LOADPLUS 51
read-only (RO) status
initial object status 71
SHRLEVEL REFERENCE option 87, 190
read-or-replication-only (RREPL) status 69, 71
read/write (RW) status
initial object status 69, 71
SHRLEVEL CHANGE option 87, 190, 712
READERS keyword of FORCE option 279, 712
real-time statistics 152, 467
reallocating data sets during a job 258
REBUILD pending (RBDP) status 71, 596
record format
CSV input file 127
SYSREC data sets 456
RECOVER pending (RECP) status 71, 74
recoverability of loaded table space 33, 145
recovering a DB2 object after terminating 478
RECOVERYDDN option 302, 507, 551
RECP (RECOVER pending) status 71
REDEFINE option
command 246
example 527
installation 712
performance considerations 645
with SQLAPPLY option 97
redefining data sets
pausing a job after analysis 258
REDEFINE option 246, 712
VCAT-defined 99
REFERENCE keyword of SHRLEVEL option
command option 190
example 487
how to run SHRLEVEL REFERENCE load 89
installation option 712
referenced files, as input
BLOBF field definition 338, 340
CLOBF field definition 338, 340
DBCLOBF field definition 338, 340
LOB data 120, 456
XML data 115, 456

Index 899
referential constraints 146

authorization for checking 50
checking 146, 263, 712
discards due to 218, 450
ENFORCE option 263, 712
example of checking 612
informational
command option 263
enforcing 146
installation option 712
preventing CHKP 151
restoring integrity 151
object status requirements for checking 69, 71
requirements for checking 147
restart considerations 467
with DSNUTILB load 146
See also referential integrity
referential constraints, authorization for checking
51
referential integrity 146
compared to IBM DB2 LOAD 33
considerations 146
ENFORCE option 263, 712
how LOADPLUS enforces 146
restoring 151
specifying enforcement of 263, 712
with SQLAPPLY load 95
See also referential constraints
REFP (refresh pending) status 71
refresh pending (REFP) status 71
REGION parameter 426
REGION parameter, recommendations for using 679
REGISTER option of COPY YES option
description 294
examples 520, 602, 619
registering copies
examples 520, 602, 619
specifying 294
related publications 13
REMBFCPY keyword of DDTYPE option 397, 762
REMBLCPY keyword of DDTYPE option 397, 762
REMBXCPY keyword of DDTYPE option 397, 762
REMOVEDECIMAL option 346
REMOVESIGN option
of BIGINT data type 345
of DECIMAL data type 346
of INTEGER data type 344
of SMALLINT data type 344
removing fractional portion of decimal value 346
removing sign
from BIGINT data type values 345
from DECIMAL data type values 346
from INTEGER data type values 344
from SMALLINT data type values 344
REMPFCPY keyword of DDTYPE option 397, 762
REMPLCPY keyword of DDTYPE option 397, 762
REMPXCPY keyword of DDTYPE option 397, 762
RENAME keyword
of ORIGDISP option 712
of ORIGINALDISP option 252
renaming staging data sets
process 100, 102
specifying maximum number of tasks 712
specifying when to begin 255
with ORIGDISP option 712
with ORIGINALDISP option 252
RENMMAX option 712
reordered row format considerations 143
REORG pending (REORP) status 71
REORG PLUS, loading data from archive data set of 197
reorganizing index while loading 648
REORP (REORG pending) status 71
REPLACE option
CHKP status setting 150
description 187
examples 499
object status 69
of PART option 306, 551
of TABLE option 69, 306, 478
performance considerations 664
replacing tables 602
report
applied fixes 427
error summary 222
of canceled threads 279, 436, 712
statistics 289
REPORT keyword
of BMCSTATS option 289
of DISCARDS option 222, 533, 557
requirements, system and software 49
reserved data set names 436
RESET keyword of MGEXTENT option 712
Resource Access Control Facility. See RACF
resource allocation
LOAD phase 693
PRELOAD phase 691
single-phase load 695
SQLAPPLY load 698
RESTART parameter of EXEC statement 427, 467
restart pending (RESTP) status 71
RESTART(PHASE) parameter of EXEC statement 427, 467

restarting
ACTIVE option considerations 398
after ANALYZE PAUSE 258
after EXCHANGE PAUSE 256
after PRELOAD PAUSE 208
after SWITCHTIME PAUSE 255
after x37 abend 467
allocation considerations 467
DSNUTILB load 467
duplicate unique keys 467
during DELETEFILES processing 467
during IDCAMS DEFINE processing 467
dynamic allocation considerations 84, 467
ENFORCE option considerations 467
ENUMROWS option considerations 467
FASTSWITCH option considerations 467
inadequate space on DB2 object 467
inadequate space on SYSERR 467
INDEX UPDATE option considerations 467
inline image copies 294, 467
input data (SYSREC) considerations 467
input data requirements 467
nonparticipating nonpartitioned indexes 467
object structure requirements 467
ORDER option considerations 467
parameters 427
partition number considerations 467
partition-by-growth table space considerations 427, 467
real-time statistics consideration 467
referential constraint checking considerations 467
single-phase load considerations 467
SKIPIX option considerations 467
SORTOUT data sets 467
SQLAPPLY option considerations 96, 467
statistics considerations 467
SYSDISC data sets 467
SYSERR data sets 467
SYSUT1 data sets 467
two-phase load considerations 467
restarting after failure 467
restoring referential and table check integrity 151
RESTP (restart pending) status 71
restrictive statuses 71
RESUME option

description 183
DISCARDS option limitation 218
object status 71
performance considerations 664
recovery considerations 160
See also LOAD RESUME YES and LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY
RETCODE option 208
retention period 421, 762
RETPD option 421, 762
retry attempts, limiting
determining stopped status 712
drain processing 712
High-speed Apply requests 190, 712
SQL statements 712
RETRY keyword of FORCE_AT option 279, 712
RETRY option 279
RETRY parameter of DYNALOC option 783
RETRY_DELAY option 279
retrying drain 279
retrying dynamic allocation of SORTWK 783
return codes
designating
on discards 223, 712
on discards with pause 208, 712
on reaching discard limit 224, 712
on zero rows loaded 225, 712
determining final 158
differences with IBM DB2 LOAD 33
from user exits 827
hierarchy 158
REUSE option 249
RI. See referential integrity
RI keyword of ENFORCE option 263, 712
RO (read-only) status
initial object status 69, 71
SHRLEVEL REFERENCE option 87, 190, 712
ROUND option
of BIGINT data type 345
of DECIMAL data type 346
of FLOAT data type 347
of INTEGER data type 344
of SMALLINT data type 344
ROUND_CEILING keyword of
DECFLOAT_ROUNDMODE option 284
ROUND_DOWN keyword of
DECFLOAT_ROUNDMODE option 284
ROUND_FLOOR keyword of
DECFLOAT_ROUNDMODE option 284
ROUND_HALF_DOWN keyword of
DECFLOAT_ROUNDMODE option 284
ROUND_HALF_EVEN keyword of
DECFLOAT_ROUNDMODE option 284
ROUND_HALF_UP keyword of
DECFLOAT_ROUNDMODE option 284
ROUND_UP keyword of
DECFLOAT_ROUNDMODE option 284
rounding 284, 379
routines, user exit
assembler 828
C 847
COBOL II 839
data type conversion 355
DB2 152
examples 826
how LOADPLUS invokes 152
invoking MODESET for 668
LE C 854
LE COBOL 839
program language 395, 712
return codes 827
specifying 395, 712
variables for DSN patterns 414
with FORMAT BMCUNLOAD option 134
row change timestamp columns 134, 324
row format considerations 143
row ID indexes 206
row length, determining average 258, 653
row-level security 162
ROWID data type
description 352
loading in a worklist environment 134
with FORMAT BMCUNLOAD option 132, 134
with VALUE option 324
RREPL (read-or-replication-only) status 69, 71
RRF. See reordered row format
rules
command syntax 163
comparison 316, 712, 819
concatenation in DSN pattern 414
input data types 379
RULES option
column name restrictions 316
comparison operators 819
constants 316
description 712
differences 819
error conditions with RULES=BMC 823
error conditions with RULES=STANDARD 822
example 514
processing order for WHEN, NULLIF, and
DEFAULTIF options 821
with FORMAT BMCUNLOAD option 137
with FORMAT CSV option 128
running BMC utilities concurrently 66, 804
running LOADPLUS 466
RW (read/write) status
initial object status 69, 71
SHRLEVEL CHANGE option 87, 190, 712

S

SAMPLE keyword of ANALYZE option 258, 653
sample LOADPLUS jobs 483
sample user exits 826
sampling
for statistics 293, 712
to estimate cardinality 258, 653
SAS/C runtime library 431
scale, specifying 379
SCAN keyword of ANALYZE option 258, 653
scanning, controlling 258, 653
SCC (DB2 Solution Common Code) 47, 49, 431
schema validation 57
SDATA parameter on IBM SDUMPX macro 712
SDUMP option 712
SDUMPX macro, IBM 712
secondary authorization IDs 50
secondary extents 712
secondary templates, switching to 411, 762
security 50
exit 51
LOADPLUS 51
verification 51
segmented table spaces
using TABLE REPLACE option 306
with FASTSWITCH 253, 712
SELECT authority
LOADPLUS 51
SELECT privileges 50
sequential data sets, extended 762
allocating 404
shadow data sets. See staging data sets
SHORTMEMORY option 239, 712
SHR, data set disposition 59
SHRLEVEL 798
SHRLEVEL CHANGE option
command option 190
c onsiderations for use 91
example 569
how to run with LOAD REPLACE option 91
how to run with LOAD RESUME YES PART
REPLACE 91
how to run with LOAD RESUME YES
SQLAPPLY option 93
installation option 712
See also LOAD RESUME YES SHRLEVEL
CHANGE SQLAPPLY and SHRLEVEL option
SHRLEVEL option
 availability considerations 87, 678
 command 189
 considerations for use 87, 688
 examples 487, 569
 FASTSWITCH process 102
 installation 712
 object status change 72
 recovery 478
 renaming process 100
 staging data sets 98
 switching process 102
SHRLEVEL REFERENCE option
 command option 190
 considerations for use 89
 examples 487, 619
 how to run 89
 installation option 712
 See also SHRLEVEL option
 sign, removing
 from BIGINT data type values 345
 from DECIMAL data type values 346
 from INTEGER data type values 344
 from SMALLINT data type values 344
 single input file (SYSREC) 675
 single-phase load
diagram of phases 38
DISCARDS option limitation 218
duplicates consideration 663
dynamic allocation 78
dynamic data set allocation 401
dynamic data set allocation by 762
dynamic data set allocation 762
dynamic allocation 78
dynamic data set allocation 762
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
dynamic allocation 78
examples 487, 507, 583
index data set use 459
load data set use 443
performance considerations 663
process 695
recovery 478
resource allocation diagram 695
restart considerations 467
SORTOUT data set use 443
specifying 208, 712
SYSUT1 data set use 459
SIX keyword of SKIPPI X option 206
size of data sets, estimating 258, 465
SIZEPCT option
 command 410
 example 562
 installation 762
SKIPFIELD S option 304, 533, 557
SKIPPIX option
description 206
example 596
setting statuses after load 74
with SQLAPPLY option 97
skipping index build process 206, 596
sliding-scale calculation for secondary extents 712
small incremental load jobs 688
SMALLINT data type 344
SMAX option
trolling sort processing 665
description 712
interaction with LOADDN option 213, 712
performance considerations 687
SMCORE option
trolling BMCSORT 665
description 712
SMS (IBM Storage Management Subsystem)
classes 422
dynami c data set allocation 401
SMS (Storage Management Subsystem)
data sets managed by 762
dynamic data set allocation 762
unit to pass to SMS 762
volumes, specifying maximum number 762
SMS (Storage Management Subsystem) classes
data classes, SMS 762
SMS extended sequential data sets 762

Index 903
allocating 404
SMS option 401, 762
SMSUNIT option 402, 762
SNAP dump, with SQLAPPLY load 95, 435
SNAPSHOT UPGRADE FEATURE (SUF)
  description 161
  licensing 160
  referential constraints, using to check 147, 161
  using to enable zIIP processing 161
  version required 49
XBM subsystem ID 269, 712
zIIP processing 268, 712
software requirements 49
solution, Database Administration for DB2 26
sort data sets. See sort work data sets
sort processing
  BMCSORT 665
    concurrent processes 236, 665, 670
    installation options 712
    memory allocation 241, 712
    memory shortage 239
    multitasking 687
    when invoked
        COMBINED phase 695
        LOAD phase 693
        PRELOAD phase 691
        SQLAPPLY load 698
sort routine. See BMCSORT
sort work data sets (SORTWK)
  allocating 446
  allocating dynamically
    by BMCSORT or LOADPLUS 446
    DTYPE option 397, 762
    SORTDEVT option 234, 712
    SORTNUM option 235, 712
  deleting 250, 712
  description 45
  estimating size 258
  multiple 446
  specifying DD statements 446
SORTDEVT option
  command 234
  examples 533, 557
  installation 712
sorting data
  memory usage 687
  performance considerations 678
  specifying 225
sorting input data 225
SORTKEYS option 207
SORTNUM option
  description 235
  examples 533, 557
  installation 712
  performance considerations 687
SORTOUT data sets
  allocating 443
  allocating dynamically 397, 762
  calculating number 208, 443, 662
  CHANGE MANAGER considerations 443
  controlling buffer usage 656
  default ddbname, overriding 213, 712
  deleting 250, 712
  description 45
  estimating size 258
  multiple 213, 443, 670
  optimal number of 670
  overriding default ddbname 213, 712
  performance considerations 656
  single-phase load use 443
  SMS extended sequential data sets 443
  specifying DD statements 443
  tuning considerations 670
  worklist environment considerations 443
SORTWK data sets
  allocating 446
  allocating dynamically
    by BMCSORT or LOADPLUS 446
    DTYPE option 397, 762
    performance considerations 645
    SORTDEVT option 234, 712
    SORTNUM option 235, 712
  deleting 250, 712
  description 45
  estimating size 258
  multiple 446
  specifying DD statements 446
SORTWORK keyword of DTYPE option
  allocating for allocation 411
SPACE option 411
SPANNED keyword of FORMAT option 201
spaced-record format data sets 201
spatial data 162
specifying DD statements 432
specifying options module 427
specifying skipped fields 304
SQL functions, indexes created on expressions that contain 142
SQL keyword
of DRAIN_WAIT option 277
of DRNWAIT option 712
SQL statements
  deleting rows from the BMCDICT table 793
  deleting rows from the BMCHIST table 797
  deleting rows from the BMCSYNC table 803,
      811
  deleting rows from the BMCSYNC table for
      RECOVER UNLOADKEYS 803
  deleting rows from the BMXCOPY table 818
  displaying BMC utilities 791
  querying BMCHIST table 791
  querying BMXCOPY table 791
  terminating BMC utilities 791
SQLAPPLY keyword
  of SHRLEVEL CHANGE command option 190
  of SHRLEVEL installation option 712
  recommendations 678
  See also LOAD RESUME YES SHRLEVEL
      CHANGE SQLAPPLY
SQLCODE errors, discarding records for 218
SQLDELAY option 712
SQLRETRY option 712
SSID (DB2 subsystem identifier) parameter of
  EXEC statement 427
stacked tape
  with copy data sets 294
  with INDSN option 211
staging data sets
  authorization 688
  deleting 252, 712
  description and use 98
  disposition 712
  FASTSWITCH process 102
  maximum tasks to start during renaming
      process 712
  naming conventions 100, 102
  performance considerations 645
  preallocating 246
  REDEFINE option 99, 246
  rename process 100, 252, 712
  renaming 252, 255, 712
  specifying when to switch 253
  switching process 102, 255
STANDARD keyword
  of NLPCTFREE option 257
  of RULES option 712
START keyword of FORCE_AT option 279, 712
start values for POSITION option 324
state for EDITPROCs 712
statistics
  DASD MANAGER PLUS tables 289
  DB2 catalog 291
  DB2 real-time 152
  options 288
  output data set, optional 436
  performance considerations 668
  report output 289
  restart considerations 467
  sampling 293, 712
  SQLAPPLY load restriction 97
  when using FIELDPROCs 289
  with DSNUUTILB load 152, 291
STATISTICS option of IBM DB2 LOAD utility 291
status of objects 33, 68, 146
status, BMC utilities 791
STEP name variable 414, 762
STEPLIB DD statement 431
STOGROUP-defined data sets
  redefining 246
  using DSNZPARM values 253
stop pending (STOPP) status 712
STOP@CMT option 712
STOPDELAY option 712
STOPP (stop pending) status 712
STOPRETRY option 645, 712
StopX37 product 55
storage above the bar 55
storage classes, SMS 422
storage classes, SMS keywords
  STORCLAS option 762
Storage Management Subsystem. See SMS
storage, virtual 55, 672
STORCLAS option 422, 762
STRIP, translating TRIM to 61
striped data sets 162, 443, 459
structure, table, when using FORMAT
  BMCUNLOAD option 134
SUBBYTE field 383
subset of partitions in single image copy 712
substitution characters 205, 218, 450
SUBSTR SQL function 142
subsystem ID
  parameter of EXEC statement 427
  variable 414, 762
  XBM 269, 712
SUF. See SNAPSHOT UPGRADE FEATURE
summary of changes 16
SUP keyword of UXSTATE option 712
supported DB2 versions 49
switch process for staging data sets
  description 102
  FASTSWITCH option 253, 712
  specifying when to begin 255
  SWITCHTIME option 255
  See also FASTSWITCH process
switching, template 411, 762
SWITCHTIME option 255, 467, 619
symbolic variables, for DSN pattern
  concatenation 414
  description 414, 762
SYNC option 97
  keywords
    SYNC option 262
syntax
  diagrams 168
  examples 483
  option descriptions 182
  rules 163
SYSALLDA unit name 762
SYSARC data set of REORG PLUS, loading data from 197
SYSDISC data sets
  abends, x37 467
  allocating 453
  allocating dynamically 397, 762
  CHANGE MANAGER considerations 453
  default ddname, overriding 216, 712
  deleting 250, 712
  description 45
  estimating size 258
  expiration date 420, 762
  GDG names 80, 414, 762
  inadequate space 467
  limiting GDG versions 422, 762
  overriding default ddname 216, 712
  retention period 421, 762
  SMS extended sequential data sets 453
  specifying DD statements 453
  worklist environment considerations 453
SYSIBM.SYSCOPY table
  duplicates in 294
  registration information 436
SYSIBM.SYNDEXSPACESTATS table 152
SYSIBM.SYSEQUENCES table 51
  MAXASSIGNEDVAL field 111
  reserving cache 273
  SELECT privileges 50
  updating 111
SYSIBM.SYSEQUENCESDEP table 51
SYSIBM.SYSEQUENCESDEP table, SELECT
  privileges on 50
SYSIBM.SYSTABLESPACESTATS table 152
SYSDCIN data sets
  data sets, LOADPLUS
    IDCAMS (SYSDCIN) 462
    SYSDCIN 462
  DD statements
    SYSDCIN 462
  default ddname, overriding 249, 712
  description 45
  IDCAMS input data sets (SYSDCIN)
    specifying and using 462
  redefining data sets
    VCAT-defined 462
  SHRLEVEL option
    SYSDCIN data set 462
    specifying and using 462
SYSSIN data sets
  description 45
  specification and usage 454
SYSSMAP data sets
allocating dynamically 397, 762
default ddname, overriding 284, 712
description 45, 454
difference from IBM DB2 LOAD utility 33
expiration date 420, 762
GDG names 80, 414, 762
limiting GDG versions 422, 762
overriding default ddname 284, 712
retention period 421, 762
SYSMAP keyword of DDTYPE option 397, 762
SYSPRIN2 data sets
  changing message levels 427
description 45
  specification and usage 455
  worklist environment considerations 455
SYSPRINT data sets
  changing message levels 427
description 45
  specification and usage 455
SYSREC data sets
  allocating 456
  allocating dynamically 211
  batch pipe data 139
  calculating file size 229, 712
  calculating number 456, 662
  concatenating 675
  considering 456, 670
  controlling buffer usage 656
  default ddname, overriding 212, 712
description 45
  estimating file size
    AUTOENUMROWS option 712
    AVGSIZE option 352–355
    ENUMROWS option 229
    LOBAVGPC option 287, 712
    XMLAVGSIZE option 288, 712
GDG names, using 211
INDSN option 211
multiple
  enabling multitasking 670
  performance considerations 675
  specifying 456
  specifying prefix for 212, 712
overriding default ddname 212, 712
pipe data 139, 218, 450
referencing files in 456
restart considerations 467
restrictions 456
spanned-record format 201
specifying DD statements 456
specifying format 196
specifying referenced files in 456
VBS 201
  with DSNUTILB load 59
system authorization
  verification 51
system dump
  excluding abend codes 712
  generating 712
system period start and end columns 138, 306, 324
system requirements 49
system setup 49
system-period temporal tables
  control card changes 132, 138
  restrictions 187, 306
SYSUT1 data sets
allocating 459
allocating dynamically 397, 762
CHANGE MANAGER considerations 459
controlling buffer usage 656
default ddname, overriding 215, 712
deleting 250, 712
description 45
estimating size 258
LOB requirements 459
multiple 459, 670
overriding default ddname 215, 712
performance considerations 656, 670
single-phase load use 459
SMS extended sequential data sets 459
specifying DD statements 459, 670
tuning considerations 656, 670
worklist environment considerations 459
XML requirements 459
cardinality 258
child, how constraint checking affects status 150
compressed 538, 551
DEFINE NO 676
empty 141, 183
LOB 122
LOGGED 33
multi-table 59, 190, 712
nonpartitioned 206, 306, 436
NOT LOGGED 33, 94
organization, how SQLAPPLY load affects 95
recoverability 33, 145, 478
redefining 246
segmented 253, 306, 712
statistics 152
status 68
versioned 134
XML 115
table structure for FORMAT BMCUNLOAD option 134
TABLE=obid option 313
tables
  application-period temporal 185, 712
  clone
    exchange process for 256
    FASTSWITCH restriction 253
    statistics restriction 289, 291
  multiple
    example 514
    with SQLAPPLY load 95, 190, 712
  organized by hash 57, 712
  system-period temporal
    control card changes 132, 138
    restrictions 187, 306
  temporal
    control card changes 132, 138
    period data override 306
    restrictions 187, 306
    transaction timestamp override 306
    with INDEX UPDATE 57, 712
tables, BMC
backing up 789
BMCHIST 38, 794
BMCLGRNX 797
BMCSYNC 38, 712, 798
BMCTRANS 806
BMCUITL 38, 712, 808
BMXCOPY 811
considerations 789
determining names 791
querying 791
warnings 789
tables, BMCDICT 792
tape
devices, number of 84
with MAXTAPE 84
tape data sets 231, 446, 712
tape management systems 231
TAPEDISP option 712
tapes
changing work file disposition 712
for copies 645
number of devices 394
tasks
COMBINED phase 695, 698
LOAD phase 693
multiple, enabling 670
optimal number, specifying ANALYZE to
determine 208
performance information 679
performed by LOADPLUS 28
PRELOAD phase 691
read 669
sort, controlling 665
started, for XBM or SUF 89, 147
type controlled by MAXSORTS 236
type controlled by SMAX 712
using XBLKS option to control 669
TEMPLATE control statements 411
TEMPLATE control statements of IBM 60, 393
template switching 411, 762
temporal tables
card changes 132, 138
period data override 306
restrictions 187, 306
transaction timestamp override 306
with INDEX UPDATE 57, 712
temporary data sets
definition 85
how LOADPLUS handles 85
specifying how to handle 712
TEMPRALDATA option 712
TERM parameter of EXEC statement 427
TERMINATEDBY option of FORMAT CSV option 199
terminating a job 477
terminating BMC utilities 477, 791
threads used by LOADPLUS 56
threads, canceling
authorizations required 50
FORCE option 279, 712
FORCE_AT option 712
FORCE_RPT option 712
reporting 279, 436, 712
specifying when to cancel 279, 712
THRESHLD option 411, 762
threshold for secondary values 83
threshold for secondary values 411, 762
time
data conversions 376
data type 351
exit routines 668
formats 356
variables 414, 762
TIME data type 351
TIME keyword
of APRETLIM option 712
of APRETRYLIM option 190
time zone, timestamp
implicit time zone 273
specifying 350
timeout
number of retry attempts 712
specifying 277, 712
specifying the number of retry attempts 279
specifying the time between retry attempts 279
time between retry attempts 712
timestamp
data conversions 376
data type 350
formats 356
row change 134, 324
with time zone
implicit time zone 273
specifying 350
TIMESTAMP data type 350
TIMESTAMP WITH TIME ZONE data type 350
tokens in syntax, rules for 163
Top Secret. See CA Top Secret
TOTALPAGEPCT option 238, 712
transaction ID columns 306
transaction start columns 138, 324
TRANSACTIONIDOVERRIDE option 306
translations
  command options for 203
  example 607
  invoking DSNUTILB 712
  performance considerations 648
  processing order 383
  supported 383
TRANSPROC field 383
TRANSTAB field 383
TRIM option
  examples 487, 520
  of BINARY data type 348
  of BINARY VARYING data type 348
  of CHAR data type 338
  of VARCHAR data type 348
  of VARCHAR data type 340
TRUNCATE option
  of BINARY data type 348
  of BINARY VARYING data type 348
  of CHAR data type 338
  of VARCHAR data type 348
  of VARCHAR data type 340
truncating
  binary values 348
  character values 338, 340
  DECFLOAT values 284
  long names in messages 712
  numeric values 379
truncating long names in messages 202
TSSAMPLEPCT option 293, 712
tuning LOADPLUS
  availability 678, 688
  command options 648
  dynamic allocation of data sets 659
  high-speed data migration 675
  I/O processing 656, 672
  improving memory use 686
  installation options 645
  memory use 686
  multitasking 670
  partitioned table spaces 676
  performance 645
  single-phase processing 663
  sort processing 665, 678
two-phase load
  diagram of phases 38
  dynamic allocation 78
  LOAD phase process 693
  PRELOAD phase process 691
  specifying 208, 712
types of discards 218

U

UNCATLG keyword of TAPEDISP option 712
underutilized pages 238, 712
Unicode
  considerations 124
  constants 125
  date exit routine 152, 668
  discards, preventing erroneous 124
  example 607
  how to load 123
  invoking DSNUTILB 712
  option to support 204
  restrictions 162
  supported translations 383
  time exit routine 668
UNICODE option
  to invoke DSNUTILB for translation processing 712
  to specify Unicode format 204
uninitialized data sets, SYSREC 456
UNIQUECHECK option
  description 244
  examples 499, 551
  with DISCARDS option 218
  with ORDER option 244
  with SQLAPPLY option 97
UNIQUEINTO option
  description 305
  example 514
  performance considerations 660
unit names, for dynamic allocation 406, 762
UNIT option
  command 406
  example 562
  installation 762
  with SMSUNIT option 83, 402
UNITCNT option 407
  example 562
  installation 762
units, specifying number of 762
UNLOAD keyword of FORMAT option 199
UNLOAD PLUS, loading data from considerations 132
CSV data 127
example 588
FORMAT BMCUNLOAD syntax 197
performance 675
UNLOADKEYS entries, cleaning up 803
unsupported features of DB2 162
unused pages 237, 712
UPDATE authority 51
UPDATE keyword of INDEX option 185, 551
UPDATEDB2 Common Statistics component option 291
UPDATEDB2STATS option
description 291
examples 499, 538, 562
with DSNUTILIB load 291
with SQLAPPLY option 97
UPDATEMAXA option 274
updating indexes 185
updating MAXASSIGNEDVAL 274, 712
updating statistics
   DASD MANAGER PLUS tables 289
   DB2 catalog 291
   DB2 real-time 152
UPDMAXA option 712
UPDMAXA_AUTHID option 712
UPPER SQL function 142
USE keyword of IFALLOC option 400, 762
user abends 208
user exits
   assembler 828
   C 847
   COBOL II 839
data type conversion 355
DB2 152
examples 826
how LOADPLUS invokes 152
invoking MODESET for 668
LE C 854
LE COBOL 839
program language 395, 712
return codes 827
specifying 395, 712
variables for DSN patterns 414
with FORMAT BMCUNLOAD option 134
user ID variable 414, 762
USER keyword of UPDMAXA_AUTHID option 712
user-defined defaults 324
user-defined return codes
determining final 158
hierarchy 158
on discards 223, 712
on reaching discard limit 224, 712
pause on discards 208, 712
zero rows loaded 225, 712
user-defined variables 414, 827
USERID (user identifier) parameter of EXEC statement 427
UT status 69, 71
UTIL keyword
   of DRAIN_WAIT option 277
   of DRNWAIT option 712
UTILB_COLCCSID option 712
UTILB_NULLIX option 712
UTILID (utility identifier)
   parameter of EXEC statement 427
UTILINIT phase, primary functions 38
utility ID variable 414, 762
utility identifier (UTILID)
   parameter of EXEC statement 427
utility parameters on the EXEC statement 427
utility restrictive state, read-only access allowed (UTRO) 69, 71
utility restrictive state, read/write access allowed (UTRW) 69, 71
utility restrictive state, utility exclusive control (UTUT) 69, 71
UTILTERM phase 38, 478
UTPRINT data set
   specification and usage 464
UTPRINT data sets
   description 45
UTRO (utility restrictive state, read-only access allowed) 69, 71
UTRW (utility restrictive state, read/write access allowed) 69, 71
UTUT (utility restrictive state, utility exclusive control) 69, 71
UXSTATE option 712

V

VALIDPROCs
discards due to violations 218, 221, 450
how LOADPLUS invokes 152, 668
using 431
violations 218, 450
with DEFINE NO objects 467
with FORMAT BMCUNLOAD option 134
VALPROC keyword of DISCARDS option 221
VALUE option 324, 527
VARBINARY data type 348
VARCHAR data type
column comparisons 316
description 340
VARGRAPHIC data type 343
variable-block spanned (VBS) data sets 201
variables for DSN pattern
concatenation 414
description 414
numeric 414
setting at installation 762
user-defined 827
VCAT-defined data sets, redefining during load process 99, 462
VCATNAME variable 414, 762
verification during FORMAT BMCUNLOAD process 135
verification records 135, 218, 450
verifying
authorization 51
versioning, index and table space 134
versions of DB2 49, 427
VIO data sets 446, 712
virtual storage 672
above the bar 55
and MEMLIMIT 55
VOLCNT option 408
example 562
installation 762
volume count 408
volumes
specifying average space on each 762
specifying maximum number 762
VSAM data sets, redefining during the load 246
VTOC, removing GDG entries from 762

WHEN option
description 313
example 596
how RULES option affects 821
of DISCARDS option 221
performance considerations 665
restriction with FORMAT option 316
with hexadecimal constant 316, 596
white space, preserving 338, 340, 355
work files
allocating 432
allocating dynamically 78, 393
checks for validity 85
deleting 82, 250, 712
description of 45
discard (SYSDISC) 450
disposition (DISP parameter)
restart considerations 467
SORTOUT data sets 467
SYSDISC data sets 450
SYSERR data sets 467
SYSREC 59
SYSUT1 data sets 467
temporary data sets 85
with DELETEFILES option 250
dynamic allocation 78, 393
error (SYSERR) 453
estimating size with ANALYZE option 258, 465
index (SYSUT1) 459
load (SORTOUT) 443
mapping (SYSMAP) 454
multiple 670
prefixes 432
restart considerations 467
sort (SORTWK) 446
SORTOUT 443
SORTWK 446
specifying DD statements 432
SYSDISC 450
SYSERR 453
SYSMAP 454
SYSUT1 459
See also dynamic allocation of data sets
WORK keyword of DDTYPE option 397
WORKDDN option
command 215
examples 507, 551
installation 712
worklist environment

W
WARN keyword of FILECHK option 712
WBUFFS option 712
WEPR (write error page range) status 71
dynamic data set allocation 80
loading ROWID columns 134
restriction when deleting files 250, 712
viewing utility output 455
with dynamic data set allocation 398
WORKUNIT option 712
write claimers, canceling 279, 712
write error page range (WEPR) status 71
WTOR keyword
  of EXCHANGE option 256
  of SWITCHTIME option 255

X

XBLKS option 669, 712
XBM
  description for RESUME YES SHRLEVEL REFERENCE 161
  licensing 160
  management set 89, 147
  referential constraints, using to check 147, 161
  subsystem ID 269, 712
  using to enable zIIP processing 161
  version required 49
  zIIP processing 268, 712
XBMGROUP name 269, 712
XBMID option
  command 269
  example 612
  installation 712
XLBnnnnn data sets 436
XLPnnnn data sets 436
XML columns defined with an XML schema 57
XML data

binary 338, 340, 355
BMCSYNC table considerations 801
compression 140
considerations for loading 114
copies for 115, 436
data type 355
example 632
free space specifications, handling 33
from referenced files
  BLOB file 338, 340
  invoking DSNUTILB 57, 712
  requirements 115, 456
how to load 118
incompatible options 115
loading from spanned-record data sets 201
preserving white space
  BLOB file 338, 340
  CLOB file 338, 340
  DBCLOB file 338, 340
schema validation 57
specifying estimated size of data
  at the field level 338, 340, 355
  globally 288, 712
XML data type 355
XML indexes 57
XML option 712
XML schema 57
XMLAVGSIZE option 288, 712
XRBnnnnn data sets 436
XRPnnnn data sets 436

Y

YEAR SQL function 142

Z

z Integrated Information Processor (zIIP). See zIIP processing
z/OS Unicode Conversion Services 383
zero rows loaded
  designating return code 225, 712
  with COPYPEND YES 294
ZEROROWRC option 712
ZIIP option 268, 712
zIIP processing
enabling 161
LOADPLUS requirements 49
option to enable or disable 268, 712
requirements 49
specifying an XBM subsystem 269

ZONED keyword of DECIMAL data type 346
ZPARAM keyword of FASTSWITCH option 712