BMC Next Generation Technology Load
for DB2 for z/OS Reference Manual

Supporting
Version 12.1 of BMC Next Generation Technology Load for DB2 for z/OS

December 2016
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>
<tbody>
<tr>
<td>BMC SOFTWARE INC</td>
<td>1 713 918 8800</td>
<td>1 713 918 8000</td>
</tr>
<tr>
<td>2103 CITYWEST BLVD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOUSTON TX 77042-2827 USA</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Outside United States and Canada

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

© Copyright 2015-2016 BMC Software, Inc.

BMC, BMC Software, the BMC logo, the BMC Software logo, and other BMC marks are the exclusive properties of BMC Software, Inc. and are registered or may be registered in the U.S. and in other countries. BladeLogic and other BladeLogic marks are the exclusive properties of BladeLogic, Inc. and are registered or may be registered in the U.S. and in other countries. All other trademarks or registered trademarks are the property of their respective owners.

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

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

Restricted rights legend

U.S. Government Restricted Rights to Computer Software. UNPUBLISHED—RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES. Use, duplication, or disclosure of any data and computer software by the U.S. Government is subject to restrictions, as applicable, set forth in FAR Section 52.227-14, DFARS 252.227-7013, DFARS 252.227-7014, DFARS 252.227-7015, and DFARS 252.227-7025, as amended from time to time. Contractor/Manufacturer is BMC SOFTWARE INC, 2103 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.
## Contents

### About this book
- Related publications ................................................................. 9
- Conventions ........................................................................... 10
- Syntax diagrams ...................................................................... 10

### Chapter 1
Overview of NGT Load ................................................................. 13

### Chapter 2
Operational considerations for NGT Load ...................................... 15
- NGT Load requirements .......................................................... 15
  - NGT Load software requirements .......................................... 15
  - Required authorizations for NGT Load .............................. 15
  - Status requirements for NGT Load ...................................... 15
- Alternate OBID processing ......................................................... 16
- CHECK constraints ................................................................ 16
- Compression dictionaries ......................................................... 16
- Error handling ........................................................................ 17
- Indexes and LOAD RESUME YES ............................................ 17
- Expression-based indexes ......................................................... 17
- Integrity checking .................................................................... 18
- Object type considerations ....................................................... 18
- Partitioned table spaces with NPIs ........................................... 19
- Recovery methods ................................................................. 19
- Restart limitations ................................................................. 20
- ROWID columns .................................................................... 21
- Shadow copies for LOAD REPLACE ...................................... 22
- Sort considerations ............................................................... 23
- SYSCOPY posting .................................................................. 23
- Table space statuses during loads .......................................... 23
- Work data set (WK00001) considerations .............................. 25
- Additional restrictions ............................................................ 25
- Differences between LOADPLUS and NGT Load .................. 25

### Chapter 3
Input for NGT Load .................................................................... 27
- DD statements ....................................................................... 27
  - LODPARMS ....................................................................... 27
  - RRGPARMS ....................................................................... 27
Chapter 4 Parameters for NGT Load

Syntax rules for parameters ........................................... 75
+AFTERACC ......................................................... 76
+ALLO .................................................................. 77
+COMMIT ................................................................. 78
+DISCMGS ............................................................. 79
+FILESZ ................................................................ 79
+PAD ..................................................................... 80
+RUNSTATS ........................................................... 81
+SETCHECKPEND .................................................... 82
+SORT ................................................................... 83
+WRITETO ............................................................... 83

Chapter 5 Syntax examples

Replace data without allowing concurrent access .............. 86
Replace data while allowing concurrent read access ............ 86
Add data to an existing table space .................................. 87
Replace one partition while allowing concurrent READ access .... 87
Replace one partition without allowing concurrent access ....... 87
Replace some partitions, add data to some partitions, and leave remaining partitions available .................... 88
Add data to an existing table space while allowing read/write access .......... 88
Remove all data from a table space .................................. 89
About this book

This book contains detailed information about the associated product or products. This preface explains the special conventions that the book uses, and how to access related publications.

If applicable, the preface also summarizes the major changes included in the latest release of the product.

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 (primary center and secured center)
  - Support Central (at http://www.bmc.com/support/mainframe-demonstrations)
  - BMC Mainframe YouTube channel (https://www.youtube.com/user/BMCSoftwareMainframe)


Products with online interfaces also offer online Help via the F1 key or, for graphical user interfaces (GUIs), via a Help button.
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: `testsyst instance/fileName`

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

Syntax diagrams

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

The following guidelines provide additional information about syntax diagrams:

- Read diagrams from left to right and from top to bottom.
- A recursive (left-pointing) arrow above a stack indicates that you may choose more than one item in the stack.
- An underlined item is a default option.
- If a diagram shows punctuation marks, parentheses, or similar symbols, you must enter them as part of the syntax.
- In general, IBM commands, keywords, clauses, and data types are displayed in uppercase letters. However, if an item can be shortened, the minimum required portion might be shown in uppercase letters, with the remainder in lowercase (for example, CANcel).
- The following conventions apply to variables in syntax diagrams:
  - Variables are typically displayed in lowercase letters and are always italicized.
  - If a variable is represented by two or more words, initial capitals distinguish the second and subsequent words (for example, databaseName).
Overview of NGT Load

This chapter contains an overview of the BMC Next Generation Technology Load for DB2 for z/OS (NGT Load) product.

NGT Load provides the following features and benefits:

- Flexible data type conversions
- Extensive integrity checking on the input data, and on newly-loaded table spaces and indexes
- Optional sorting of input data
- Discard processing for conversion and duplicate index key errors
- The ability to override defined OBIDs when loading from a FORMAT (UNLOAD) input file
- Support for ROWID columns
- Support for both GENERATED ALWAYS and GENERATED BY DEFAULT identity columns
- Support for indexes defined on an expression
- Use of the IBM z Systems Integrated Information Processor (zIIP)
- Enabling LOAD REPLACE in read-only or read/write mode
- Keeping nonpartitioned indexes in read/write status when running either LOAD RESUME YES or LOAD REPLACE
- One-pass loading of multiple tables
- Quick recovery from a failed LOAD RESUME YES job
- A robust SQL-like data manipulation language
- Support for security labels
Operational considerations for NGT Load

This chapter provides information that you need to know to run NGT Load.

NGT Load requirements

This topic describes NGT Load requirements.

NGT Load software requirements

The NGT Load product requires a minimum of version 12.1.00 of the BMC DB2 Solution Common Code (SCC).

Required authorizations for NGT Load

To run NGT Load, you need the following authorizations:
- LOAD authority on the database for the object that you are loading
- SYSADM authority or DBADM authority
- Ownership of the table that you are loading

Status requirements for NGT Load

When running LOAD RESUME NO with NGT Load, the table space cannot be in any restricted status.
Alternate OBID processing

NGT Load supports an override of the defined OBID when loading from a FORMAT(UNLOAD) input file.

When you use FORMAT(UNLOAD), NGT Load compares the DB2-defined OBID with the OBID of each input row. If the OBIDs do not match, the row is not processed. If the OBID keyword is specified in the LOAD DATA statement, NGT Load compares the specified OBID value with the input row OBID and selects only matching rows for processing. Rows whose OBIDs do not match are ignored.

If you turn off OBID checking, NGT Load loads all input rows. To do this, code the OBID keyword with a value of 0. This forces NGT Load to load to a table from a DB2 system different from the one unloaded, or to load from an unload file created before a table was dropped and recreated.

**WARNING**

Be careful when you use the OBID keyword. If you specify a bad value or have not performed OBID checking, DB2 might abend.

CHECK constraints

When a null field is encountered during a CHECK constraint clause, the row is discarded. To prevent this, qualify the CHECK constraint with **OR field IS NULL**.

Compression dictionaries

The following considerations apply to using compression dictionaries with NGT Load.

LOAD REPLACE creates and uses the compression dictionary. The end-of-job summary includes a report of any compression activity. The compression dictionary build fails if there is insufficient data to build one. NGT Load loads uncompressed data with a warning (condition code of 4) and clear message.

LOAD RESUME YES uses compression dictionaries as follows:

- If a dictionary currently exists for the table space, NGT Load uses it to compress your input data.
If a dictionary does not currently exist and the table space is defined with COMPRESS YES, NGT Load builds a dictionary based on the input data and applies it to the rows being added only.

Error handling

NGT Load handles errors in a similar fashion to the IBM DB2 LOAD utility.

- If NGT Load detects a conversion error during processing, but finds no defined discard data set (as evidenced by the presence of a SYSDISC DD statement or equivalent), then processing fails.

- If NGT Load detects a limit key error during processing, processing continues.

- If NGT Load detects a duplicate key during processing, processing continues.

---

**Note**

Message NGTL903 might appear, flagging that rows have been rejected due to errors in Phase 2 of processing. For example:

```
LD900DCL NGTL903 28 ROWS REJECTED DUE TO ERRORS AT PHASE 02
```

Errors in Phase 2 of processing are duplicate keys. A load job with this message ends with a return code of 4. This is normal and does not signify a problem.

Indexes and LOAD RESUME YES

Although LOAD RESUME YES always appends data at the end of the table space, indexes on the table being loaded are rebuilt or reorganized using shadow copies with a rename at the completion of the load process.

Expression-based indexes

NGT Load supports loading tables that have an index defined on an expression. For information about supported expressions, see the *BMC Next Generation Technology General User Guide*. 

Integrity checking

NGT Load performs integrity checks on the input data, and on newly loaded table spaces and indexes. NGT Load also checks unique indexes for duplicate keys.

If NGT Load detects an inconsistency of any kind, processing stops immediately. Additional action required depends on the type of job that you are running:

- Due to the non-destructive nature of LOAD RESUME YES, LOAD REPLACE SHRLEVEL CHANGE, or LOAD REPLACE SHRLEVEL REFERENCE in NGT Load, such an inconsistency never damages your data.

- A LOAD REPLACE job without SHRLEVEL CHANGE or SHRLEVEL REFERENCE might require recovery. For more information, see “Recovery methods” on page 19.

**Note**

NGT Load does not perform referential integrity (RI) checking.

Object type considerations

The following topics describe considerations for loading specific types of objects.

Partition-by-growth table spaces

NGT Load loads partition-by-growth table spaces at the table space level only; you cannot use the PART option.

If additional partitions are required during the load process and the maximum number of partitions allowed for the table space has not been reached, NGT Load adds the required partitions up to the maximum number of partitions. NGT Load fails if additional partitions are needed and the maximum number of partitions has been reached.

Unsupported objects

NGT Load does not support the following objects:

- Catalog tables
- Encrypted tables
- Hash-organized tables
- Table spaces with clone tables
Simple multi-table table spaces
Archive tables, when you specify LOAD REPLACE

Partitioned table spaces with NPIs

The following considerations apply to nonpartitioned index (NPI) changes during the load:

- When you specify LOAD REPLACE at the table space level with either SHRLEVEL CHANGE or SHRLEVEL REFERENCE, any NPI changes for the partitions that you are loading are discarded. NGT Load collects changes for the remaining partitions and applies them after the load.

- You can keep the NPI in read/write status throughout the load process by specifying REPLACE or RESUME YES at the partition level only (on the INTO TABLE clause). NGT Load collects changes to the NPI and applies them later. Using this method, partitions that are specified as REPLACE or RESUME YES change status before processing starts; other partitions do not change status and the NPI remains in read/write status.

Recovery methods

If a job fails, the correct method of recovery depends on the type of job that you are running.

- If you are running LOAD RESUME YES SHRLEVEL CHANGE, restart the job from the last commit point during the failed load.

- If you are running any other LOAD RESUME YES, you do not need to perform recovery. Instead, perform one of the following steps:
  - Use the QUICKEXIT keyword to terminate the utility.
    The table space becomes available without the new data.
  - Use the RESTART keyword to continue the load process.

- If you are running LOAD RESUME NO, restart the job. If the table space was not empty, it will be available in read/write status.

- If you are running LOAD REPLACE, recover by using one of the following options:
  - Restart the LOAD REPLACE job.
Perform your standard method for recovery using an image copy (SHRLEVEL NONE only).

If you are loading a partitioned table space and have a mixture of RESUME YES and REPLACE on the INTO statement, perform one of the following steps:

— Use the QUICKEXIT keyword to terminate the utility.
  Partitions with a RESUME YES specification are restored to their original status. Partitions with a REPLACE specification must be recovered.

— Restart the job.
  Partitions with a RESUME YES specification are restored to their original status and the job starts from the beginning.

Restart limitations

The topic describes the consequences of job failure during various phases of the load process.

The load process involves the following eight phases:

1. Before access (STOP table space and indexes)
2. SYSREC read, table space load, and keys sort
3. Index build
4. Duplicate key removal
5. Switch phase
6. After access (START table space and indexes)
7. Final discard
8. Completion

If a failure occurs before phase 3, the entire load process becomes non-restartable.

If a failure occurs during phase 3, NGT Load restarts the load process, provided that the following conditions are met:

- No nonpartitioned index (NPI) is in R/W status during the load.
- You have not changed the table space or index status before restarting.
If a failure occurs during phase 4, the entire process becomes non-restartable.

When a non-restartable condition is detected, the NGT Load utility performs cleanup and one of the following tasks:

- If you have specified RESTART, restarts
- If QUICKEXIT is specified, terminates

**ROWID columns**

NGT Load supports ROWID columns.

NGT Load generates ROWID values unless they are provided in the input data. The following example shows a ROWID column specification:

```sql
Example

LOAD DATA INDDN SYSREC LOG NO REPLACE
INTO TABLE NGT.LOB_TABLE
( "COL01"     "COL02"     "COL03"     "COL04"     "ROWIDCOL"
  " POSITION(3:6)" " POSITION(7:15)" " POSITION(17:17)" " POSITION(19:19)" " POSITION(20:42)"
  INTEGER       CHAR(9)       CHAR(1)       CHAR(1)       ROWID
)
```

**How NGT Load loads ROWID values**

When loading ROWID columns that are defined as NOT NULL GENERATED ALWAYS, NGT Load generates ROWIDs. The LOAD statement column specification for the table must not reference the ROWID columns. If the column specification references the ROWID columns, NGT Load issues an error message and the load fails.

NGT Load also supports ROWID columns that are defined as NOT NULL GENERATED BY DEFAULT. The input load data may or may not contain values for these columns. If the input data does not contain ROWID values, the column specification must not reference the ROWID column and NGT Load generates ROWID values when loading. If the input data does contain ROWID values, the column specification must reference the ROWID column and NGT Load loads the ROWID values from the input data.
Data in UNLOAD format

When you load UNLOAD format data, the ROWID value must exist in the input data. NGT Load loads the ROWID values from the input data when the column is defined as NOT NULL GENERATED BY DEFAULT. NGT Load does not generate new ROWID values when loading data in UNLOAD format.

If the ROWID column is defined as NOT NULL GENERATED ALWAYS, NGT Load does not support the loading of UNLOAD format data. In this case, NGT Load issues an error message and the load fails.

Shadow copies for LOAD REPLACE

For LOAD REPLACE, NGT Load supports loading data to a shadow copy.

You can enable this functionality in one of the following ways. For each of these methods, NGT Load creates and loads shadow copies of the table space and index, but does not touch the original table space and index. The original objects remain unchanged until the load process finishes and the data sets are renamed or switched.

- Specify SHRLEVEL REFERENCE on your LOAD command.
  With this option, NGT Load places the table space or partition in read-only status before the start of the load process. For partial loads, NPIs are in read/write status.

- Specify SHRLEVEL CHANGE on your LOAD command.
  With this option, NGT Load places the table space or partition in read/write status before the start of the load process. For partial loads, NPIs are in read/write status.

- Specify the +WRITETO(B) parameter in //LODPARMS and either specify or default to SHRLEVEL NONE on your LOAD command.
  With this parameter, NGT Load places all affected objects in read-only status before the start of the load process.

For more information, see “SHRLEVEL” on page 38 and “+WRITETO” on page 83.
Sort considerations

NGT Load sorts input data only if you specify the +SORT(YES) parameter or the ORDER YES syntax option. NGT Load sorts the data based on the definition of the clustering index.

When you are loading more than 200 million records, you must use the +FILESIZE keyword to provide an estimated number of thousand rows to be loaded. NGT Load then sorts the records without the need for any external sort products. It sorts data based on the clustering index key to generate a cluster ratio of 100 after LOAD REPLACE finishes.

LOAD RESUME YES SHRLEVEL NONE also sorts data based on the clustering index key if you have specified +SORT(YES), but the cluster ratio might be less than 100 owing to other data already in the table space.

NGT Load always sorts the keys for the indexes and produces reorganized indexes (except when you perform insert loading with LOAD RESUME YES SHRLEVEL CHANGE).

SYSCOPY posting

NGT Load posts loads in SYSIBM.SYSCOPY. Note the following considerations:

■ The load is posted as LOG(NO).

■ Unless you specify COPYDDN or COPY YES, NGT Load sets COPY pending status for the table space being loaded if the environment dictates.

■ NGT Load sets COPY pending for the following types of load jobs:
  — LOAD REPLACE
  — LOAD RESUME YES without SHRLEVEL CHANGE specified
  — LOAD RESUME NO

Table space statuses during loads

The following table shows the DB2 object statuses during various load scenarios.

Note

For either LOAD REPLACE or LOAD RESUME YES, all objects relating to the table specified in the INTO clause change status before the load process begins.
### Table 1: Table space statuses during load scenarios

<table>
<thead>
<tr>
<th>Load scenario</th>
<th>Object status</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD DATA INDDN SYSREC01 RESUME YES SHLLEVEL CHANGE INTO TABLE creator.tableName</td>
<td>RW (insert load)</td>
</tr>
<tr>
<td>LOAD DATA INDDN SYSREC01 RESUME YES INTO TABLE creator.tableName</td>
<td>STOP</td>
</tr>
<tr>
<td>LOAD DATA INDDN SYSREC01 RESUME YES SHLLEVEL CHANGE INTO TABLE creator.tableName PARTS 2,4,6 REPLACE INTO TABLE creator.tableName PARTS 1,3,5 RESUME YES</td>
<td>PARTS 1, 3, 5 STOP PARTS 2, 4, 6 RW Other partitions and NPIs RW</td>
</tr>
<tr>
<td>LOAD DATA INDDN SYSREC01 REPLACE SHLLEVEL REFERENCE INTO TABLE creator.tableName</td>
<td>RO</td>
</tr>
<tr>
<td>LOAD DATA INDDN SYSREC01 REPLACE SHLLEVEL Change INTO TABLE creator.tableName</td>
<td>RW</td>
</tr>
<tr>
<td>LOAD DATA INDDN SYSREC01 REPLACE SHLLEVEL CHANGE INTO TABLE creator.tableName PART 2 REPLACE</td>
<td>PART 2 RW NPIs RW</td>
</tr>
<tr>
<td>With +WRITETO(A) NGT Load parameter: LOAD DATA INDDN SYSREC01 RESUME NO REPLACE INTO TABLE creator.tableName</td>
<td>STOP</td>
</tr>
<tr>
<td>With +WRITETO(B) NGT Load parameter: LOAD DATA INDDN SYSREC01 RESUME NO REPLACE INTO TABLE creator.tableName</td>
<td>RO</td>
</tr>
<tr>
<td>With +WRITETO(A) NGT Load parameter: LOAD DATA INDDN SYSREC01 INTO TABLE creator.tableName PART 2 REPLACE</td>
<td>PART 2 STOP Other partitions and NPIs RW</td>
</tr>
<tr>
<td>With +WRITETO(B) NGT Load parameter: LOAD DATA INDDN SYSREC01 INTO TABLE creator.tableName PART 2 REPLACE</td>
<td>PART 2 RO Other partitions and NPIs RW</td>
</tr>
<tr>
<td>With +WRITETO(A) NGT Load parameter: LOAD DATA INDDN SYSREC01 INTO TABLE creator.tableName PART 2 REPLACE (field specifications)</td>
<td>PARTS 1-6 STOP Other partitions and NPIs RW</td>
</tr>
<tr>
<td>With +WRITETO(B) NGT Load parameter: LOAD DATA INDDN SYSREC01 INTO TABLE creator.tableName PART 2 REPLACE (field specifications)</td>
<td>PARTS 1, 3, 5 STOP PARTS 2, 4, 6 RO Other partitions and NPIs RW</td>
</tr>
</tbody>
</table>
Work data set (WK00001) considerations

The following considerations apply to the NGT Load work file:

- During the load process, the size required for the WK00001 data set is double the size of the SYSREC file. LOAD uses this disk space for internal sort processing. For more details about the allocation of the WK00001 work data set, see the topic about the XSUTDBMG automation control point in the NGT Automation Reference Manual.

- NGT Load deletes this work file after the load completes.

Additional restrictions

NGT Load has the following additional restrictions:

- NGT Load does not support XML data or data structures.
- NGT Load does not support data from the IBM SQL/DS product.

Differences between LOADPLUS and NGT Load

Review this topic if you are an existing user of LOADPLUS and need to understand how NGT Load differs from the functionality you have been using.

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Differences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax</td>
<td>For syntax differences, see “LOAD syntax that is no longer valid” on page 71.</td>
</tr>
<tr>
<td>zIIP offload processing</td>
<td>NGT Load does not require the BMC EXTENDED BUFFER MANAGER for DB2 (XBM) product or its SNAPSHOT UPGRADE FEATURE for DB2 (SUF). The NGT subsystem handles zIIP offload processing.</td>
</tr>
</tbody>
</table>
| Data sets: dynamic allocation | With NGT Load, you can dynamically allocate your discard and copy data sets by either using the appropriate automation control point or by using the OUTPUT command.  
For more information about the automation control point, see the BMC Next Generation Technology General User Guide. For more information about the OUTPUT command, see “OUTPUT command” on page 31. |
<p>| Statistics             | NGT Load does not update the BMC DASD MANAGER PLUS for DB2 statistics tables.                                                                |</p>
<table>
<thead>
<tr>
<th>Functionality</th>
<th>Differences</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMAT</td>
<td>NGT Load does not provide a BMC-internal format for moving data more quickly between NGT Unload and NGT Load. NGT Load does not support loading data in spanned-record format.</td>
</tr>
<tr>
<td>SHRLEVEL</td>
<td>Although NGT Load LOAD RESUME YES SHRLEVEL CHANGE uses SQL INSERT processing just as LOADPLUS did, NGT Load does not require the use of the BMC High-speed Apply Engine product. When you specify LOAD RESUME YES SHRLEVEL CHANGE with NGT Load, you cannot make a copy during the load job. You cannot run LOAD RESUME YES SHRLEVEL REFERENCE with NGT Load.</td>
</tr>
<tr>
<td>Installation options</td>
<td>Instead of using the DOPTs installation options, NGT Load uses configuration parameters to provide default processing values. You can also specify these parameters in your SYSIN.</td>
</tr>
<tr>
<td>Referential constraints</td>
<td>NGT Load does not perform referential constraint checking.</td>
</tr>
<tr>
<td>User-defined return codes</td>
<td>For NGT Load, you can define your own return codes by using the XSUTTERM automation control point.</td>
</tr>
<tr>
<td>Copies</td>
<td>When you specify LOAD RESUME YES SHRLEVEL CHANGE with NGT Load, you cannot make a copy during the load job. NGT Load always registers all copies. NGT Load does not make inline copies.</td>
</tr>
<tr>
<td>Invoking DSNUTILB</td>
<td>NGT Load does not invoke DSNUTILB.</td>
</tr>
<tr>
<td>Processing phases</td>
<td>NGT Load does not have a two-phase processing option.</td>
</tr>
<tr>
<td>XML data</td>
<td>NGT Load does not support XML data.</td>
</tr>
<tr>
<td>LOB data</td>
<td>NGT Load supports LOB data only if you specify SHRLEVEL CHANGE. NGT Load does not currently support spanned-record format.</td>
</tr>
<tr>
<td>Compression dictionaries</td>
<td>NGT Load does not use the KEEPDICTIONARY option to determine whether to keep or build a dictionary. For more information about how NGT Load handles compression dictionaries, see “Compression dictionaries” on page 16.</td>
</tr>
<tr>
<td>RBA/LRSN conversion</td>
<td>NGT Load enables you to convert from basic format to extended format RBA/LRSN values. LOADPLUS did not provide this ability.</td>
</tr>
</tbody>
</table>
Input for NGT Load

This chapter describes the DD statements, SYSIN input (syntax statements), and keywords specific to NGT Load.

DD statements

The following topics describe the DD statements used with NGT Load.

LODPARMS

This DD is required only to override the NGT Load parameters specified at installation time.

You may specify LODPARMS in a data set or in the JCL. For example:

```
//LODPARMS DD DISP=SHR,DSN=NGT.LOAD.PARMS
```

or

```
//LODPARMS DD *
```

RRGPARMS

This DD is required to specify certain parameters from BMC Next Generation Technology Reorg for DB2 for z/OS (NGT Reorg) that also pertain to NGT Load.

You may specify RRGPARMS in a data set or in the JCL. For example:

```
//RRGPARMS DD DISP=SHR,DSN=NGT.REORG.PARMS
```
or

//RRGPARMS DD *

UTLPARMS

This DD is required only to override the NGT utilities global parameters specified at installation time.

You may specify UTLPARMS in a data set or in the JCL. For example:

//UTLPARMS DD DISP=SHR,DSN=NGT.UTIL.PARMS

or

//UTLPARMS DD *

For information about global parameters, see the *BMC Next Generation Technology General User Guide*.

RUNSTATS

This DD is an optional DD statement for RUNSTATS reports.

You can send this DD statement to an output file or to a SYSOUT output class. For example:

//RUNSTATS DD DISP=SHR,DSN=NGT.RUNSTATS.REPORT

or

//RUNSTATS DD SYSOUT=* 

SYSREC

This DD names the input file and is required.

You can override this DD name using the INDDN option on the LOAD statement. The data in the input file must match the format described by the LOAD statement.

The following statement is an example of a SYSREC DD statement

//SYSREC DD DISP=SHR,DSN=NEW.LOAD.INPUT.FILE
As an alternative to specifying an input file name in your DD statement, you can use the OUTPUT command. For more information about this command, see the *BMC Next Generation Technology General User Guide*.

**SYSDISC**

This DD names the discard data set.

You must allocate this DD to activate discarding. You can override this DD name using the DISCARDDN option on the LOAD statement. The attributes LRECL, RECFM, and DSORG of this data set must match those of the input file (SYSREC).

The following statement is an example of a SYSDISC DD statement:

```
//SYSDISC DD DISP=SHR, DSN=DISCARDS.OUTPUT.FILE
```

As an alternative to specifying a discard data set name in your DD statement or the DISCARDDN option, you can dynamically allocate the data set by using the OUTPUT command. For more information about this command, see the *BMC Next Generation Technology General User Guide*.

**Note**

If processing detects a conversion error and a SYSDISC DD statement (or equivalent DD, as specified by DISCARDDN) is not specified, processing fails.

**NGTAUTO**

If you are using automation control points with your job, you can specify an NGTAUTO DD statement to override the default data set (specified during configuration or with the +NGTAUTO1 or +NGTAUTO2 parameter).

This statement should reference a partitioned data set containing the NGT automation control points. For example:

```
//NGTAUTO DD DISP=SHR, DSN=NGT.AUTOMATN.CTRL.PTS
```

For more information about NGT automation control points, see the *BMC Next Generation Technology Automation Reference Manual*. 
SYSIN

This DD is required.

Specify syntax statements for the utility with the SYSIN DD statement, as follows:

```plaintext
//SYSIN DD *
......(syntaxStatement)
......(syntaxStatement)
......(syntaxStatement)
```

If you specify the OUTPUT command, it must be placed before your LOAD statement.

For more information about syntax for NGT Load, see NGT Load syntax diagrams on page 32.

NGT Load syntax overview

The following sections describe the LOAD command.

You can also specify the OUTPUT command to dynamically allocate certain data sets. For more information, see “OUTPUT command” on page 31.

Command syntax rules for NGT utilities

The following general rules apply to NGT utilities command syntax:

- In an NGT utility command, the utility recognizes the following comment characters:
  - An asterisk (*) in column 1
    The utility ignores the entire line.
  - Two consecutive hyphens in a line
    The utility ignores the remainder of the line.
  - REXX-like comments (starting with /* and ending with */)
    The utility ignores the comment text, including /* and */. You can specify these comments anywhere within the statement and you can nest these comments.

- 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, the utility 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.

Except where indicated in the syntax diagram, you cannot specify the same command option more than once. Duplicate command options cause the utility to terminate.

In the syntax diagrams in the following pages, underlined options indicate default options.

### OUTPUT command

You can use the OUTPUT command, with your NGT Load SYSIN, to dynamically allocate discard and copy data sets, and to build input data set names. The following considerations apply to the OUTPUT command with NGT Load:

- For input data sets, the OUTPUT command must reference existing data sets.
- You specify the OUTPUT command before the LOAD statement.

For details, see the following information:

- For details about the OUTPUT command and its options, see the *BMC Next Generation Technology General User Guide*.

- For information about using this command for input data sets, see “INDDN ddname” on page 35.

- For information about using this command for discard data sets, see “DISCARDDN ddname” on page 46.

- For information about using this command for copy data sets, see “COPYDDN(lp,lb)” on page 53, “RECOVERYDDN(rp,rb)” on page 54, and “COPY” on page 55.
NGT Load syntax diagrams

This topic contains NGT Load syntax diagrams. Descriptions of the syntax keywords follow these diagrams.
INTO-spec

- INTO TABLE 
  - creator.
  - tableName
  - PART
    - PARTS
      - partitionNumber
      - partitionNumber
  - OBID(integer)
  - IGNOREFIELDS NO
  - SKIPFIELDS YES
  - REPLACE
  - RESUME NO
  - WHEN
  - WHERE 
    - WHERE clause
  - field selection
  - WHEN
  - WHERE 
    - WHERE clause
  - field specification
**Basic options**

This topic describes basic NGT Load options.

**DATA**

This keyword is optional.
INDDN \textit{ddname}

INDDN specifies the DD name that will be used to read the input data. The default DD name is SYSREC.

If you specify this keyword, you must allocate the specified DD name. The DD name must reference an existing data set and the data set may be of fixed or variable format (blocked is also valid) and must be readable by the basic sequential access method (BSAM).

As an alternative to specifying an input data set name in your DD statement, you can use the OUTPUT command. The OUTPUT command must reference an existing input data set. For more information about this command, see the \textit{BMC Next Generation Technology General User Guide}.

COPYDICTIONARY \textit{(partitionNumber)}

COPYDICTIONARY tells NGT Load to copy an existing compression dictionary from the specified partition to other partitions on a partitioned table space.

NGT Load copies the current compression dictionary from the specified partition and uses it to compress the input data for the partitions being replaced. This option enables you to copy a compression dictionary to an empty partition in which a compression dictionary would not normally have been built.

You can specify any valid partition number for the table space, as long as you are not replacing that partition. The default value is 1.

You can use the COPYDICTIONARY option when both of the following conditions exist:

- You are loading a partitioned or range-partitioned table space.
- You specify RESUME NO at the table space level and PART \textit{integer} REPLACE statements.

The following considerations apply to the COPYDICTIONARY option:

- You cannot specify COPYDICTIONARY with the following options:
  - LOAD PART \textit{integer} RESUME
  - LOAD RESUME YES
- In addition, the COPYDICTIONARY keyword is incompatible with the following options:
  - KEEPDICTIONARY at the table space or partition level
— REPLACE at the table space level
— LOAD RESUME NO

- The REPLACE keyword copies only the compression dictionary to partitions being replaced that have the COMPRESS YES attribute. A valid dictionary must exist for the partition being copied.

**RESUME**

RESUME tells NGT Load whether to add new data to the existing data in the table space.

---

**Note**

You can mix RESUME YES and REPLACE in one LOAD statement for different partitions by repeating the INTO TABLE clause for different field specifications. For more information, see the description for RESUME on the INTO TABLE clause (“RESUME” on page 58).

---

**YES**

RESUME YES tells NGT Load to add new data to the existing data in the table space.

**NO**

RESUME NO functions differently depending on whether you specify RESUME NO, or default to it.

Defaulting to RESUME NO (specifying neither LOAD RESUME nor LOAD REPLACE) tells NGT Load to load data into the table space if it is empty. NGT Load terminates if it encounters data while reading the table space.

If you specify RESUME NO explicitly, NGT Load does not check whether the table space is empty. In this case, you can include RESUME and REPLACE specifications at the partition level on the INTO TABLE clause (just as you can when you specify RESUME YES).

**REPLACE**

When you specify REPLACE, NGT Load resets the table space and index spaces to empty before loading any data.
RBALRSN_CONVERSION

RBALRSN_CONVERSION enables you to convert the DB2 RBAs and LRSNs between basic (6-byte) and extended (10-byte) format. The RBALRSN_CONVERSION keyword must immediately follow the REPLACE keyword.

The default behavior depends on the UTILITY_OBJECT_CONVERSION subsystem parameter. If you do not specify RBALRSN_CONVERSION and have set UTILITY_OBJECT_CONVERSION to BASIC, then RBALRSN_CONVERSION defaults to BASIC. If you have set UTILITY_OBJECT_CONVERSION to NOBASIC or EXTENDED, then RBALRSN defaults to EXTENDED.

You can specify one of the following keywords:

BASIC

Converts from extended format to basic format. If you specify BASIC and set the UTILITY_OBJECT_CONVERSION subsystem parameter to NOBASIC, the object fails the reorganization.

EXTENDED

Converts from basic format to extended format. If you specify EXTENDED and set the UTILITY_OBJECT_CONVERSION subsystem parameter to BASIC, the object fails the reorganization.

NONE

Does not perform conversions. If you specify NONE, have set the UTILITY_OBJECT_CONVERSION subsystem parameter to NOBASIC, and the object is in basic format, the object fails the reorganization.

ROWFORMAT

ROWFORMAT converts the row format when you are performing a LOAD REPLACE. You can specify one of the following keywords:

RRF

Converts from basic row format (BRF) to reordered row format (RRF).

BRF

Converts from RRF to BRF.
PREFORMAT

PREFORMAT tells NGT Load whether to preformat unused table space pages before loading the data.

If you specify PREFORMAT without a value, NGT Load assumes PREFORMAT YES.

YES

Preformatting consists of writing zero pages from the high-used RBA to the high-allocated RBA. When you specify PREFORMAT YES, NGT Load preformats all pages up to the high-allocated RBA.

NGT Load terminates if you specify PREFORMAT YES with RESUME YES SHRLEVEL CHANGE.

NO

If you specify PREFORMAT NO, NGT Load does not preformat the unused pages.

SHRLEVEL

SHRLEVEL specifies the extent to which applications can concurrently access a table space or partition during an NGT Load job.

Discard processing differs slightly when working with RESUME YES SHRLEVEL NONE rather than RESUME YES SHRLEVEL CHANGE. If loading a file containing duplicate keys, RESUME YES SHRLEVEL NONE does not load any of the key values (including the original value), while RESUME YES SHRLEVEL CHANGE loads only the first key value of the set of duplicates. The following tables illustrate this processing:

Table 2: RESUME YES SHRLEVEL NONE

<table>
<thead>
<tr>
<th>Row number</th>
<th>Key value</th>
<th>Status at the end of the job</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>'ABC123'</td>
<td>DISCARDED</td>
</tr>
<tr>
<td>2</td>
<td>'ABC123'</td>
<td>DISCARDED</td>
</tr>
<tr>
<td>3</td>
<td>'ABC123'</td>
<td>DISCARDED</td>
</tr>
<tr>
<td>4</td>
<td>'ABC123'</td>
<td>DISCARDED</td>
</tr>
<tr>
<td>5</td>
<td>'ABC123'</td>
<td>DISCARDED</td>
</tr>
</tbody>
</table>
Rows loaded: 0

Table 3: RESUME YES SHRLEVEL CHANGE

<table>
<thead>
<tr>
<th>Row number</th>
<th>Key value</th>
<th>Status at the end of the job</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>'ABC123'</td>
<td>LOADED</td>
</tr>
<tr>
<td>2</td>
<td>'ABC123'</td>
<td>DISCARDED</td>
</tr>
<tr>
<td>3</td>
<td>'ABC123'</td>
<td>DISCARDED</td>
</tr>
<tr>
<td>4</td>
<td>'ABC123'</td>
<td>DISCARDED</td>
</tr>
<tr>
<td>5</td>
<td>'ABC123'</td>
<td>DISCARDED</td>
</tr>
</tbody>
</table>

Rows loaded: 1

You can specify one of the following keywords:

**NONE**

SHRLEVEL NONE (the default) specifies that applications have no concurrent access to the table space or partition.

You cannot use SHRLEVEL NONE when loading LOB data.

+WRITETO(B) overrides SHRLEVEL NONE.

**CHANGE**

SHRLEVEL CHANGE specifies that applications can concurrently read from and write to the table space or partition into which NGT Load is loading data. You can specify SHRLEVEL CHANGE with either LOAD RESUME YES or LOAD REPLACE.

**LOAD RESUME YES SHRLEVEL CHANGE**

This specification tells NGT Load to use SQL to insert rows into the specified tables. Therefore, switching or change of status does not occur.

If you specify RESUME YES SHRLEVEL CHANGE, you cannot specify any of the following options:

- RESUME NO
- REPLACE
- KEEPDICTIONARY
- LOG NO
- ENFORCE NO
- COPY YES
■ COPYDDN
■ RECOVERYDDN
■ PREFORMAT

LOAD REPLACE SHRLEVEL CHANGE

This specification tells NGT Load to write the loaded data to shadow data sets, keeping the original data sets intact during the load process. After the data is loaded, NGT Load performs rename or fastswitch processing and deletes the original data sets.

**WARNING**
This option fully replaces the data in the table space or partition, including any inserts, updates, or deletes to the original object that occur during the load process.

The following considerations apply to LOAD REPLACE SHRLEVEL CHANGE:

■ You cannot specify COPY YES.

■ If you specify PART, the following processes take place:
  
  — NGT Load replaces the specified partitions and empties all other partitions. NGT Load does not apply changes to any of the partitions that occur during the load process.
  
  — During the load, NPIs are in RW status. NPI changes that occur during the load for the partitions that are being loaded are discarded. NPI changes that occur during the load for the remaining partitions are applied after the load.

■ NGT Load drains the objects before the switch phase. You can control the time-out values for this drain by using the +QRETRY parameter or the DRAIN_WAIT syntax option.

**REFERENCE**

SHRLEVEL REFERENCE specifies that applications can concurrently read from the table space or partition into which NGT Load is loading data. You can specify SHRLEVEL REFERENCE only with LOAD REPLACE.

NGT Load writes the loaded data to shadow data sets, keeping the original data sets intact during the load process. After the data is loaded, NGT Load stops the objects briefly to perform rename or FASTSWITCH processing, then deletes the original data sets.

The following considerations apply to SHRLEVEL REFERENCE:
If you specify PART, NPIs are in RW status during the load. NPI changes that occur during the load for the partitions that are being loaded are discarded. NPI changes that occur during the load for the remaining partitions are applied after the load.

If you do not specify COPY YES or COPYDDN, NGT Load places the table space in copy pending status after the load.

If you do specify COPY YES or COPYDDN, the copy is a SHRLEVEL REFERENCE copy.

**OBID(integer)**

Use the OBID option to specify the OBID of the table that you are loading.

This option overrides the row selection process used when loading rows using FORMAT UNLOAD. Specify a valid OBID value in decimal format. The following considerations apply to the OBID option:

- If you do not specify this option, the input data OBID must match the OBID of the table that you are loading.

- NGT Load discards any input rows that do not match.

- If you specify OBID(0), NGT Load does not perform OBID checking and all input rows are processed.

---

**WARNING**

Use the OBID keyword with care. If you specify a bad value or OBID checking is not performed, DB2 might abend.

**KEEPDICTIONARY**

KEEPDICTIONARY is available for compatibility with the IBM DB2 LOAD utility command. NGT Load disregards this option.

---

**Note**

If you are using an existing LOADPLUS job, be aware that this option does not function the same way that it did in LOADPLUS.

For more information about compression dictionaries, see “Compression dictionaries” on page 16.
LOG

LOG is available for compatibility with the IBM DB2 LOAD utility command.

NGT Load terminates if you specify LOG NO with RESUME YES SHRLEVEL CHANGE. Otherwise, NGT Load disregards the LOG option.

Note
If you are using an existing LOADPLUS job, be aware that this option functions a little differently than it did with LOADPLUS.

FLOAT

FLOAT specifies that NGT Load should expect the designated format for floating point numbers. You can specify one of the following values:

S390

(default) FLOAT S390 specifies that NGT Load should use floating point numbers in IBM System/390 hexadecimal floating-point (HFP) format. S390 is the format in which DB2 stores floating-point numbers.

IEEE

FLOAT IEEE specifies that NGT Load should use floating-point numbers in IEEE binary HFP format. When specified, NGT converts binary floating-point (BFP) data to HFP format as the data is being loaded into the DB2 table. If a conversion error occurs during the process, NGT places the record in the discard file.

Input data format

Specify one of the following options to indicate the expected format of the input data format.

EBCDIC

(default) EBCDIC specifies that the input data file is in EBCDIC format.
ASCII

ASCII specifies that the input data file is in ASCII format. The ASCII option does not affect numeric, date, time, or timestamp internal formats.

UNICODE

UNICODE specifies that the input data file is in Unicode format. The UNICODE option does not affect numeric, date, time, or timestamp internal formats.

CCSID (sbcsmixed,dbcscolumn)

CCSID specifies one or more coded character set identifiers (CCSIDs) for the input file.

You can specify one or more CCSIDs as follows:

- The first value specifies the CCSID for SBCS data that is found in the input file.
- The second value specifies the CCSID for mixed DBCS data.
- The third value specifies the CCSID for DBCS data.

If any of these positional values is specified as 0 or omitted, the CCSID of the corresponding data type in the input file is assumed to be the same as the installation default CCSID, based on the input data:

- If the input data is EBCDIC, the omitted CCSIDs are assumed to be the EBCDIC CCSIDs that are specified at installation.
- If the input data is ASCII, the omitted CCSIDs are assumed to be the ASCII CCSIDs that are specified at installation.
- If the input data is Unicode, the default CCSID values are the Unicode CCSIDs that are specified at system installation.

If the CCSIDs of the input data file do not match the CCSIDs of the table that is being loaded, NGT Load converts the input data to the table CCSIDs before it is loaded.

FORMAT

FORMAT identifies the input data format.

You can specify one of the following keywords:
UNLOAD

FORMAT UNLOAD is compatible with the IBM DB2 FORMAT UNLOAD. In the UNLOAD format, all variable length records are expanded to the maximum length. Each input row must contain a 6-byte record prefix.

**Note**
If you are using an existing LOADPLUS job, be aware that this option might not function in the same way that it did in LOADPLUS.

DELIMITED

FORMAT DELIMITED specifies that the input data file is in a delimited format. When data is in a delimited format, all fields in the input data set are character strings or external numeric values. Each column in a delimited file is separated from the next column by a column delimiter character.

For each of the delimiter types that you can specify, you must ensure that the delimiter character is specified in the code page of the source data. You can specify the delimiter character as either a character or a hexadecimal constant. For example, to specify `#` as the delimiter, you can specify either `COLDEL '#'` or `COLDEL X'23'`. For Unicode and ASCII encodings, BMC recommends that you specify the delimiter character in the utility statement as a hexadecimal constant; otherwise, the result is unpredictable.

The following restrictions apply to FORMAT DELIMITED:

- You cannot specify the same character for more than one type of delimiter (COLDEL, CHARDEL, or DECPT).
- When you specify FORMAT DELIMITED, you cannot specify the CONTINUEIF or WHEN options.
- NGT Load ignores any specified POSITION statements within the field specification.

You can specify the following keywords with FORMAT DELIMITED:

**COLDEL coldel**

COLDEL specifies the column delimiter that is used in the input file. The default value is a comma (,). For ASCII and UTF-8 data, the default value is X'2C'. For EBCDIC data, the default value is X'6B'.
CHARDEL chardel

CHARDEL specifies the character string delimiter that is used in the input file. The default value is a double quotation mark ("). For ASCII and UTF-8 data, the default value is X'22'. For EBCDIC data, the default value is X'3F'.

The following considerations apply to the CHARDEL keyword:

- To delimit character strings that contain the character string delimiter, repeat the character string delimiter where it is used in the character string. LOAD interprets any pair of character delimiters that are found between the enclosing character delimiters as a single character. For example, the phrase "FRED"S JOB" is interpreted as "FRED"S JOB". The LOAD utility recognizes these character delimiter pairs for CHAR, VARCHAR, and CLOB fields only.

- Character string delimiters are required only when the string contains the delimiter character. However, you can put the character string delimiters around other character strings.

- Data unloaded in delimited format by NGT Unload includes character string delimiters around all character strings.

DECPT decpt

DECPT specifies the decimal point character that is used in the input file. The default value is a period (.). The only allowed delimiters for DECPT are a period (.) or comma (,) (or their equivalents in ASCII or Unicode).

---

Note
The default decimal point character in a delimited file is a period (X'2E' in an ASCII or Unicode UTF-8 file).

---

ENFORCE

Specify ENFORCE to tell NGT Load how to handle DB2 check constraints.

NGT Load terminates if you specify ENFORCE with RESUME YES SHRLEVEL CHANGE.

You can specify one of the following options:

CONSTRAINTS

Specify ENFORCE CONSTRAINTS to tell NGT Load to enforce DB2 check constraints. NGT Load does not support referential integrity checking.
NO

Specify ENFORCE NO to tell NGT Load to not enforce DB2 check constraints.

ERRDDN

ERRDDN is available for compatibility with the IBM DB2 LOAD utility command. NGT Load disregards this option and writes all error messages to SYSPRINT.

Note
If you are using an existing LOADPLUS job, be aware that this option does not function as it did in LOADPLUS.

MAPDDN

MAPDDN is available for compatibility with the IBM DB2 LOAD utility command. NGT Load disregards this option.

DISCARDDN ddname

DISCARDDN identifies the DD name for the discard data set.

If you specify DISCARDDN and allocate the named data set, discard processing is enabled. If you specify DISCARDDN but have not allocated the data set, NGT Load issues an error message and the load terminates.

As an alternative to specifying a discard data set name in your DD statement, you can use the OUTPUT command. For more information about this command, see the BMC Next Generation Technology General User Guide.

DISCARDS(integer)

The DISCARDS option specifies the number of error records allowed before a load terminates.

You can specify 0 or any positive integer. For example, if you specify DISCARDS(1), the load terminates after the first record has been discarded. If you specify DISCARDS(0), NGT Load might discard the entire input data set.
CONTINUEIF(condition)

CONTINUEIF tells NGT Load whether the current logical input record continues with the next physical input record. This enables NGT Load to concatenate multiple physical records into a single logical input record before converting the data to a DB2 input record.

The two numbers, start:end, indicate the starting and ending columns of the continuation indicator in the physical input record. If the condition (the comparison of the continuation indicator with the string) is true, NGT Load concatenates the next physical input record.

You can specify any relational operator in the condition.

MAXCONN(integer)

MAXCONN controls the maximum number of threads performing inserts concurrently in a RESUME YES SHRLEVEL CHANGE load.

Specify a value greater than one to obtain multiple concurrent threads that perform inserts. If you do not specify a value, you obtain one insert thread.

This option also limits the number of concurrent converters when performing a normal (non-insert) load. For a normal load, specify a number to limit the number of data converters running concurrently. Converters use substantial CPU, but run entirely on zIIP engines (if available). If you do not specify a value, you obtain the maximum number of converters.

MAXTASKS(integer)

MAXTASKS enables NGT Load to read multiple concatenated input files in parallel.

If you have a very large amount of data to load and it is in multiple concatenated files, use this option to specify how many of those files you want NGT Load to read in parallel.

SORTDEVT

SORTDEVT is available for compatibility with the IBM DB2 LOAD utility command. NGT Load disregards this option.
**SORTNUM**

SORTNUM is available for compatibility with the IBM DB2 LOAD utility command. NGT Load disregards this option.

**WORKDDN**

WORKDDN is available for compatibility with the IBM DB2 LOAD utility command. NGT Load disregards this option.

*Note*

If you are using an existing LOADPLUS job, be aware that this option does not function as it did in LOADPLUS.

**ORDER YES**

The ORDER YES option tells NGT Load to sort the table rows in clustering index order.

This option overrides a value of NO for the NGT Load +SORT parameter. For more information about this parameter, see “+SORT” on page 83.

The following considerations apply to ORDER YES:

- ORDER YES affects only new rows that you are loading. For a RESUME YES job, NGT Load does not mix new rows with existing rows.

- 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.

- If you specify the LOADPLUS option ORDER YES ASSOCIATE, NGT Load terminates. NGT Load ignores the LOADPLUS options ORDER NO and ORDER PRESORTED.

**REDEFINE**

This option controls whether NGT Load deletes and redefines the VSAM data sets for the table space and index spaces that are participating in the load.

NGT Load terminates if you specify this option with SHRLEVEL CHANGE.
YES

REDEFINE YES tells NGT Load to delete and redefine the VSAM data set for each
DB2 object before loading it. NGT Load also defines any additional VSAM data sets
that might be required.

NGT Load redefines the table space that you are loading as well as any participating
indexes.

NO

REDEFINE NO tells NGT Load to reuse the existing VSAM data sets for the table
space and index spaces that are participating in the load.

REUSE

NGT Load uses this option for compatibility with the IBM DB2 LOAD utility
command syntax.

When you specify REUSE, NGT Load functions as if you specified REDEFINE NO. If
you specify REUSE and REDEFINE YES, NGT Load uses the last keyword that it
finds in the command string.

NGT Load terminates if you specify REUSE with RESUME YES SHRLEVEL
CHANGE.

DELETEFILES

The DELETEFILES option tells NGT Load whether to delete the INFO and OUT files
after the load completes successfully. This option overrides the NGT utilities
+DELETEOUTPUT parameter.

Note
NGT Load always deletes the work data sets (WK files) after the load completes.

NO

DELETEFILES NO tells NGT Load not to delete any INFO or OUT files.
YES

DELETEFILES YES tells NGT Load to delete all corresponding INFO and OUT files after the load completes successfully.

CHECKPEND

CHECKPEND tells NGT Load whether to set dependent table spaces to CHECK pending (CHKP) status. This option overrides the NGT Load +SETCHECKPEND parameter.

YES

After loading, NGT Load sets CHECK pending (CHKP) status for the table space if appropriate.

NO

After loading, NGT Load does not set CHKP status for the table space or any of its children.

WARNING

If you are considering specifying CHECKPEND NO, ensure that you understand the risks and ramifications to the integrity of your RI data structure.

UPDATEDB2STATS

UPDATEDB2STATS tells NGT Load whether to update statistics in the DB2 catalog. DB2 uses these statistics to determine the access paths that the DB2 optimizer selects.

This option overrides the NGT Load +RUNSTATS parameter. For more information about this parameter, see “+RUNSTATS” on page 81.

NO

UPDATEDB2STATS NO, which is the default, tells NGT Load not to update statistics in the DB2 catalog.
YES

UPDATEDB2STATS YES tells NGT Load to update statistics in the DB2 catalog according to the values in the STTPARMS DD or the NGT utilities configuration. The STTPARMS DD is the location in which the parameters for BMC Next Generation Technology Stats for DB2 for z/OS (NGT Stats) are specified. For more information, see the BMC Next Generation Technology Stats for DB2 for z/OS Reference Manual.

Restrictions

UPDATEDB2STATS YES is subject to the following restrictions:

■ NGT Load does not gather statistics during LOAD RESUME YES processing.

■ NGT Load computes aggregate statistics using catalog values for partitions that are not processed. To collect statistics for all partitions and ensure that NGT Load computes meaningful values when performing a partial load, run NGT Stats on the entire table space when, or immediately after, the table space has loaded.

■ NGT Load skips non-indexed tables.

Additional considerations

The following additional considerations apply to UPDATEDB2STATS YES:

■ NGT Load updates column statistics for every column of every index.

■ You can also have NGT Load collect frequency distributed statistics can also be collected. The NGT Stats parameters (from the STTPARMS DD) control frequency statistics collection. (If the STTPARMS DD does not supply these parameters for the load job, NGT Load obtains them from the NGT configuration.)

■ For partitioned table spaces, NGT Load does not update table statistics unless there is an index on the partitioning columns or a DPSI defined.

■ For a list of which DB2 catalog statistics you can update, see the BMC Next Generation Technology Stats for DB2 for z/OS Reference Manual.

IDENTITYOVERRIDE

The IDENTITYOVERRIDE option tells NGT Load 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, NGT Load assumes IDENTITYOVERRIDE YES.
YES

IDENTITYOVERRIDE YES tells NGT Load to load identity column values from the input file when the identity column is defined as GENERATED ALWAYS.

NGT Load does not update the MAXASSIGNEDVAL column of the SYSIBM.SYSSEQUENCES table when loading identity column values from the input file.

NO

IDENTITYOVERRIDE NO tells NGT Load not to load identity column values from the input file when the identity column is defined as GENERATED ALWAYS. In this case, NGT Load generates values and updates the MAXASSIGNEDVAL column of SYSIBM.SYSSEQUENCES.

DRAIN_WAIT

The DRAIN_WAIT option specifies how long, in seconds, NGT Load can hold a drain. You can specify any integer value with this option from 1 through 1800.

Note

NGT Load terminates if you specify any of the values allowed in LOADPLUS other than an integer (for example, DRAIN_WAIT UTIL).

You can specify this option with any of the following types of load jobs:

- LOAD RESUME YES PART REPLACE
- LOAD REPLACE SHRLEVEL CHANGE
- LOAD PART n REPLACE
- LOAD PART n RESUME

This option overrides the value of the first subparameter of the NGT utilities +QRETRY parameter. For more information about the +QRETRY parameter, see the BMC Next Generation Technology General User Guide.

If it cannot drain all of the objects within the time period specified by DRAIN_WAIT, NGT Load completes the following process:

1. Releases the drains that it has obtained so far
2. Tries again to drain the objects for the number of times that you specify in the second subparameter of the +QRETRY parameter
APCOMMIT

APCOMMIT specifies the number of records to insert before performing a commit during a LOAD RESUME YES SHRLEVEL CHANGE job.

Specify an integer to tell NGT Load the maximum number of rows to insert before performing a commit. You can specify a value of -1 to tell NGT Load to perform a commit after all rows have been inserted.

This option overrides the NGT Load +COMMIT parameter.

Image copy options

This topic describes the options that enable image copies during a load job.

COPYDDN(lp,lb)

COPYDDN tells NGT Load to produce a primary and, optionally, a backup copy to the indicated local copy destinations.

The first value specifies a local primary (lp) copy; the second value specifies a local backup (lb) copy. If you specify COPYDDN, you must specify at least a local primary copy.

These values do not specify a DD name; they signify the intention to produce a copy to this destination. The actual unit name and options for the copy destinations are defined by the one of the following methods. The value of the +OVERRIDEOUTPUT parameter determines the method that NGT Load uses.

- The XCPYDYNM automation control point (either directly or by using the NGTTAPE command)

- The OUTPUT command
  You must place the OUTPUT command before the LOAD statement in your SYSIN.

For more information about XCPYDYNM, see the BMC Next Generation Technology Automation Reference Manual. For more information about the OUTPUT command and the +OVERRIDEOUTPUT parameter, see the BMC Next Generation Technology General User Guide.

The following considerations apply to COPYDDN:
You cannot specify COPYDDN if you specify RESUME YES SHRLEVEL CHANGE.

You must specify the OUTPUT command, or the XCPYDYNM automation control point must be available to provide the image copy data set name for each image copy.

The parentheses are optional if you specify only one destination name (for example, COPYDDN LP1); the parentheses are required if you specify two DD names.

COPYDDN specified before the INTO TABLE can have two positional parameters, for example COPYDDN (A,B). COPYDDN coded after the INTO TABLE can have only one positional parameter, for example COPYDDN(A).

The following table shows the partitions that NGT Load copies:

### Table 4: Commands to copy partitions

<table>
<thead>
<tr>
<th>Command</th>
<th>Partition copied</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD DATA INDDN SYSREC REPLACE COPYDDN(A) INTO TABLE A.B</td>
<td>Whole table</td>
</tr>
<tr>
<td>LOAD DATA INDDN SYSREC RESUME YES COPYDDN(A) INTO TABLE A.B</td>
<td>Only the partitions that are loaded</td>
</tr>
<tr>
<td>LOAD DATA INDDN SYSREC RESUME YES INTO TABLE A.B REPLACE PART X COPYDDN(A)</td>
<td>Only the partitions that are loaded</td>
</tr>
<tr>
<td>LOAD DATA INDDN SYSREC RESUME YES COPYDDN(A) INTO TABLE A.B REPLACE PART X</td>
<td>Only the partitions that are loaded</td>
</tr>
<tr>
<td>LOAD DATA INDDN SYSREC INTO TABLE A.B REPLACE PART X COPYDDN(A)</td>
<td>Only partition X</td>
</tr>
</tbody>
</table>

### RECOVERYDDN(rp,rb)

RECOVERYDDN tells NGT Load to produce a primary and, optionally, a backup copy to the indicated remote copy destinations.

The first value specifies a remote primary (rp) copy; the second value specifies a remote backup (rb) copy.

These values do not specify a DD name; they signify the intention to produce a copy to this destination. The actual unit name and options for the copy destinations are defined by one of the following methods. The value of the +OVERRIDEOUTPUT parameter determines the method that NGT Load uses.
The XCPYDYNM automation control point (either directly or by using the NGTTAPE command)

The OUTPUT command
You must place the OUTPUT command before the LOAD statement in your SYSIN.

For more information about XCPYDYNM, see the BMC Next Generation Technology Automation Reference Manual. For more information about the OUTPUT command and the +OVERRIDEOUTPUT parameter, see the BMC Next Generation Technology General User Guide.

The following requirements and considerations apply to the RECOVERYDDN option:

You cannot specify RECOVERYDDN if you specify RESUME YES SHRLEVEL CHANGE.

You must specify the OUTPUT command, or the XCPYDYNM automation control point must be available to provide the image copy data set name for each image copy.

The parentheses are optional if you specify only one destination name. The parentheses are required if you specify two DD names.

**COPY**

The COPY option tells NGT Load whether or not to produce a copy of the table space or selected partitions during the load.

**NO**

COPY NO (the default) tells NGT Load to not make a copy of the table space.

*Note*
NGT Load ignores the LOADPLUS COPYPEND option.

**YES**

If you specify COPY YES, NGT Load 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 options. NGT Load creates a standard DB2 image copy.
The unit name and options for the copy destinations are defined by one of the following methods. The value of the \texttt{+OVERRIDEOUTPUT} parameter determines the method that NGT Load uses.

- The XCPYDYNM automation control point (either directly or by using the \texttt{NGTTAPE} command)
- The \texttt{OUTPUT} command

You must place the \texttt{OUTPUT} command before the \texttt{LOAD} statement in your \texttt{SYSIN}.

For more information about XCPYDYNM, see the \textit{BMC Next Generation Technology Automation Reference Manual}. For more information about the \texttt{OUTPUT} command and the \texttt{+OVERRIDEOUTPUT} parameter, see the \textit{BMC Next Generation Technology General User Guide}.

The following considerations apply to the \texttt{COPY} option:

- You cannot specify \texttt{COPY} if you specify \texttt{RESUME YES SHRLEVEL CHANGE}.
- You must specify the \texttt{OUTPUT} command, or the XCPYDYNM automation control point must be available to provide the image copy data set name for each image copy.
- NGT Load registers all copies with DB2 in the \texttt{SYSIBM.SYSCOPY} table. If you specify the \texttt{LOADPLUS REGISTER NONE} or \texttt{REGISTER \texttt{ddname}} options, NGT Load terminates.
- NGT Load ignores any \texttt{LOADPLUS INLINE} specification.

\section*{INTO TABLE options}

Use the \texttt{INTO TABLE} clause to tell NGT Load where to load data and which data to load.

NGT Load enables you to load multiple tables in the same table space by using a single load statement that supplies multiple \texttt{INTO TABLE} clauses. Each \texttt{INTO TABLE} clause can reference a different column specification clause and a different \texttt{WHERE} clause.

Follow \texttt{INTO TABLE} with the name of the table to load. Optionally, you can fully qualify the table name with the creator ID. NGT Load updates all indexes defined on the table.

Use the following additional options with your \texttt{INTO TABLE} clause.
PART (partitionNumber)

Use PART to specify which partitions you want to load.

You can specify a single partition number or a series of partition numbers, either as individual numbers or as ranges of partition numbers. Parentheses are optional. The following examples illustrate the use of this option:

```
PART 1
PART(2:5)
PART(1,4,10:20,33:99)
```

Separate individual partition numbers with commas, and separate individual ranges of partitions with colons. You can combine individual partitions and ranges in a single PART statement.

OBID(integer)

Use the OBID option on the INTO TABLE statement to specify the OBID of your input file (either SYSREC or the file that you specify with INDDN). By specifying this OBID, you map the OBID of the input file to the OBID of the table that you are loading.

This option is meaningful only when you specify FORMAT UNLOAD. NGT Load ignores this option when loading from input files that are not in UNLOAD format.

IGNOREFIELDS or SKIPFIELDS

IGNOREFIELDS specifies whether NGT Load should skip fields in the input data set that do not correspond to columns in the target table. Alternatively, you can specify the LOADPLUS option SKIPFIELDS.

For specific examples of the use of the IGNOREFIELDS keyword, see “Use IGNOREFIELDS with the VALUE option to convert data” on page 91.

NO

(default) NO tells NGT Load to not skip any fields.

YES

YES tells NGT Load to skip fields in the input data set that do not correspond to columns in the target table.
For example, specifying YES is useful if each input record contains a variable-length field, followed by variable-length data that you do not want to load, and then data that you do want to load. The variable-length field prevents you from using the POSITION keyword to skip over the variable-length data that you do not want to load. By specifying IGNOREFIELDS YES or SKIPFIELDS YES, you can assign a field specification to the variable-length data that you do not want to load. Giving it a name that is not one of the table column names enables NGT Load to skip the field without loading it.

Use this option with care. For example, if you load a column and misspell the name, NGT Load skips fields.

When you specify YES, you can use a VALUE clause for dynamic data manipulation of all columns that allow the VALUE specification. For example, you can use a VALUE clause to simulate the STRIP functionality:

```
TEMP01 POSITION(1:30) CHAR, COL01 VALUE(STRIPE(TEMP01, B, X'00'))
```

This is the functional equivalent of using the following IBM DB2 LOAD statement:

```
COL01 POSITION(1:30) CHAR STRIP
```

This technique enables NGT Load to perform complex data manipulations dynamically. For more information about the VALUE clause, see “VALUE” on page 60.

### REPLACE

Specify REPLACE on the INTO TABLE statement to reset the specified partitions to empty before loading any data.

You can use REPLACE and RESUME YES in a single NGT Load step by specifying multiple INTO TABLE statements. For an example, see “Replace some partitions, add data to some partitions, and leave remaining partitions available” on page 88.

If you specify LOAD RESUME YES SHRLEVEL CHANGE and a mix of PART RÉPLACE and PART RESUME YES, NGT Load stops the RESUME YES partitions during the load.

### RESUME

Specify RESUME YES on the INTO TABLE statement to add new data to the existing data in the specified partitions.
RESUME NO is available for compatibility with the IBM DB2 LOAD utility command. NGT Load disregards this option.

You can use RESUME YES and REPLACE in a single NGT Load step by specifying multiple INTO TABLE statements. For an example, see “Replace some partitions, add data to some partitions, and leave remaining partitions available” on page 88.

If you specify LOAD RESUME YES SHRLEVEL CHANGE and a mix of PART REPLACE and PART RESUME YES, NGT Load stops the RESUME YES partitions during the load.

WHEN and WHERE

Specify either WHEN or WHERE to tell NGT Load when to accept an input record for processing.

To specify a standard WHEN clause, use the field selection options as shown in the field selection diagram in “NGT Load syntax diagrams” on page 32.

Alternatively, you can specify the general purpose WHERE clause instead of the standard WHEN clause. The WHERE clause references fields by name only and can use arithmetic or character expressions on named fields. For more information about the WHERE clause, see the *BMC Next Generation Technology General User Guide*.

You can perform arithmetic directly on named fields in the input record if the input is in UNLOAD format. If the input is in other formats, you need the S370 function to convert the data to useable numeric form.

Note

If you are using an existing LOADPLUS job, ensure that your WHEN clause meets NGT Load criteria for a WHEN clause. If it does not meet these criteria, your job step terminates.

Field specification

The field specification defines the format and data type characteristics of the input data.

If you include a field specification, the input fields can be in any position or sequence. Also, the clauses that make up the entire field specification do not have to be in order. The following additional considerations apply to field specifications:
If you omit the field specification, the input data must be in the format specified by the FORMAT option. (For example, if you specify FORMAT UNLOAD, the input must be in UNLOAD format.) If you do not specify the FORMAT option, the input must be in DSNTIAUL format and must not have any nullable fields.

If you include a field specification, you must include all columns that are defined as NOT NULL without defaults. Any column excluded from this clause because it is nullable receives a null value. Any column excluded from this clause because it is defined as NOT NULL WITH DEFAULT receives a default value.

As in DB2 LOAD, if the input file has unpadded variable length fields, any fields that follow are misaligned. To avoid this issue, specify PAD when unloading the data.

**VALUE**

You can use the VALUE specification to assign a constant value to a column. The constant assigned to a column must be of the same type as the column.

The VALUE specification accepts any SQL expression. Note the following restrictions:

- You cannot specify both VALUE and POSITION.
- You cannot specify VALUE on a column that determines the partitioning limit value or a clustering index key column.

The following examples show constants for different data types:

**Example**

CHARACTER, VARCHAR

Values for character columns must be a quoted string. For example:

\[\text{VALUE('a string')}\]

**Example**

INTEGER, SMALLINT, BIGINT

Values for integer and small integer columns must be numeric. For example:

\[\text{VALUE(12)}\]
\[\text{VALUE(3456)}\]
Example

**DECIMAL**

Values for decimal columns must be numeric. For example:

- `VALUE(12)`
- `VALUE(34.02)`

Example

**FLOAT, DECFLOAT**

Values for float columns must be numeric.

- `VALUE(12)`
- `VALUE(-459.2E-20)`

Example

**DATE**

Values for DATE columns must be a date or the special constant CURRENT DATE.

- `VALUE(CURRENT DATE)`
- `VALUE(dd.mm.yyyy)`
- `VALUE(mm/dd/yyyy)`
- `VALUE/yyyy-mm-dd)`

Example

**TIME**

Values for TIME columns must be a date or the special register CURRENT TIME.

- `VALUE(CURRENT TIME)`
- `VALUE(hh:mm:ss)`
- `VALUE(hh.mm.ss)`

Example

**TIMESTAMP**

Values for TIMESTAMP columns must be a date or the special register CURRENT TIMESTAMP.

- `VALUE(CURRENT TIMESTAMP)`
- `VALUE/yyyy-mm-dd-hh.mm.ss.nnnnnn)`

Note

NGT Load supports local date and time option.
**NULLIF(fieldSelection)**

NULLIF conditionally specifies whether a table column should receive the input data value (if provided) or a DB2 null value. You cannot specify NULLIF clauses for columns that are defined as NOT NULL.

The condition that you specify uses the same syntax as the field selection specified on a WHEN clause. For more information, see the field selection diagram in “NGT Load syntax diagrams” on page 32.

**DEFAULTIF(fieldSelection)**

DEFAULTIF conditionally specifies whether a table column should receive the input data value (if provided) or a DB2 default value. You can define DEFAULTIF clauses for any column, regardless of whether the column has the WITH DEFAULT attribute.

The condition that you specify uses the same syntax as the field selection specified on a WHEN clause. For more information, see the field selection diagram in “NGT Load syntax diagrams” on page 32.

The following table describes default values that NGT Load generates when a DEFAULTIF condition is true:

<table>
<thead>
<tr>
<th>Column type</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Character or graphic</td>
<td>Blank</td>
</tr>
<tr>
<td>Numeric</td>
<td>0</td>
</tr>
<tr>
<td>Date</td>
<td>Current date</td>
</tr>
<tr>
<td>Time</td>
<td>Current time</td>
</tr>
<tr>
<td>Timestamp</td>
<td>Current time stamp</td>
</tr>
</tbody>
</table>

NGT Load calculates default values only once at the start of the load process.

**Data type keywords**

Use the following data type keywords to identify the input data format.
The data type of the input data does not have to correspond to the data type of the target DB2 table column if the conversion between the input data type and the target data type is allowed, as described in “Data type conversions” on page 68.

**CHARACTER**

CHARACTER defines a fixed-length character field. You can optionally specify a length attribute with CHARACTER. You can abbreviate this keyword as CHAR.

NGT Load accepts all characters in a CHARACTER field. NGT Load does not remove leading or trailing blanks.

You can optionally specify MIXED (indicating mixed character set data) or BIT (indicating BIT data).
**VARCHAR**

VARCHAR defines a varying-length character field. A *two-byte length field* that tells NGT Load the length of the column must precede the data. The length field specifies the data length of the data, not including the two-byte length field or any padding characters.

You can optionally specify MIXED (indicating mixed character set data) or BIT (indicating BIT data).

NGT Load does not remove leading or trailing blanks.

**GRAPHIC**

GRAPHIC defines a fixed-length graphic field. NGT Load supports the GRAPHIC data type in the same way that DB2 supports it.

**VARGRAPHIC**

VARGRAPHIC defines a variable-length graphic field. NGT Load supports the VARGRAPHIC data type in the same way that DB2 supports it.

**SMALLINT**

SMALLINT indicates a two-byte signed integer.

**INTEGER**

INTEGER indicates a four-byte signed integer. You can abbreviate this keyword as INT.

**INTEGER EXTERNAL**

INTEGER EXTERNAL defines a field that contains a character representation of integer data. The data may have a plus or minus sign preceding or following the data.

Leading or trailing blanks or blanks between the sign and the number are allowed. NGT Load removes them during conversion to internal data types.

You can optionally specify a length attribute with INTEGER EXTERNAL. You can abbreviate EXTERNAL as EXT.
**BIGINT**

BIGINT indicates an eight-byte binary number. Negative numbers are in two’s complement notation.

**DECIMAL or DECIMAL PACKED**

DECIMAL or DECIMAL PACKED indicates the input data is in a packed decimal format that conforms to the rules of packed decimal numbers. NGT Load considers DECIMAL and DECIMAL PACKED data types to be identical, and treats them the same for conversion purposes. You may abbreviate DECIMAL as DEC.

**DECIMAL ZONED**

DECIMAL ZONED indicates that the input data is in the zoned format. For example:

\[ ZN...ZNSN \]

ZN stands for zone and number. The zone character may be any EBCDIC character and is ignored for the purposes of conversion. The last byte, SN, stands for sign and number. The sign character (highest four bits) must be in the range A-F. Characters A, C, E, and F represent a positive sign; characters B and D represent a negative sign.

DECIMAL ZONED is the only numeric format for external data in which leading and trailing blanks are not removed.

---

**Example**

NGT Load converts the following zoned string:

\[ X'40C1F253D4' \] or character 'A2.M'

to the following character string. Note that NGT Load removes the leading 0 (X'40') during the conversion process.

'\-1234'

---

**DECIMAL EXTERNAL**

All rules for data formats that apply to INTEGER EXTERNAL also apply to DECIMAL EXTERNAL, with the exception that the numeric portion of the input field can contain a decimal point.

The \{(length, scale)\} qualifier is optional. The length field designates the input field length. The scale field designates the implied scale of the decimal point in the input data. A decimal point in input data overrides the implied scale specification of the DEC EXT designation.
**Example**

If an input field has been defined in the field specification clause as DEC EXT (8,3) and the value is 12345678, NGT Load treats the value as 12345.678 for the purposes of conversion.

If a different row of input for the same field has a value of 765.4321, NGT Load ignores the implied scale factor of 3 and uses a scale factor of 4.

---

**FLOAT, REAL, or DOUBLE PRECISION**

These data types indicate four- or eight-byte signed data that conforms to the format of S/370-S/390 floating point numbers. A designation of REAL denotes a length of four bytes; a designation of DOUBLE PRECISION denotes a length of eight bytes. You can shorten DOUBLE PRECISION to DOUBLE.

**FLOAT EXTERNAL**

FLOAT EXTERNAL allows the most flexibility of input formats of numeric data. The rules for data formats that apply to DECIMAL EXTERNAL apply also to FLOAT EXTERNAL, and the input number may be in scientific or exponential notation. Blanks can be anywhere except in the middle of the numeric portion of the data.

NGT Load considers the examples to be valid FLOAT EXTERNAL values:

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>+123E20</td>
</tr>
<tr>
<td>-459.2e -20</td>
</tr>
<tr>
<td>100.3-</td>
</tr>
<tr>
<td>-77080.105829 E+25</td>
</tr>
</tbody>
</table>

**DECFLOAT**

DECFLOAT indicates a 64-bit or 128-bit decimal floating-point number. The length value must be either 16 or 34. If the length is 16, the number is in 64-bit decimal floating-point number format. If the length is 34, the number is in 128-bit decimal floating-point format. If the length is not specified, the number is in 128-bit decimal floating-point format.

**DECFLOAT EXTERNAL**

DECFLOAT EXTERNAL indicates a character string that represents a number. The format is an SQL numeric constant. If you do not specify a length, NGT Load determines the string length from the POSITION specification. If you do not specify a length or POSITION, NGT Load uses the DECFLOAT default length.
**DATE EXTERNAL**

DATE EXTERNAL indicates a character string that represents a date. The default length is 10 characters. Date formats may be in any of the following formats:

- `dd.mm.yyyy`
- `mm/dd/yyyy`
- `yyyy-mm-dd`

NGT Load calculates the input data format for DATE at each occurrence of a DATE field in the input data. Therefore, you may mix formats in your input data.

**TIME EXTERNAL**

TIME EXTERNAL indicates a character string that represents a time of day. The default length is 8 bytes. Time formats may be in any of the following formats:

- `hh.mm.ss`
- `hh:mm:ss`
- `hh:mm AM`
- `hh:mm PM`

NGT Load calculates the TIME input data format at each occurrence of a TIME field in the input data. Therefore, you may mix formats in your input data.

**TIMESTAMP EXTERNAL**

TIMESTAMP EXTERNAL indicates a character string that represents a time stamp. The default length is 26 bytes. Valid lengths are 16 through 26 bytes. Time stamps have the following format:

- `yyyy-mm-dd-hh.mm.ss.nnnnnn`

The `nnnnnn` microseconds value is optional. If not specified, it defaults to 000000. If the `nnnnnn` value has trailing zeroes, you can omit them.

You can omit leading zeroes from the month (`mm`), day (`dd`), and hour (`hh`) subfields.

**TIMESTAMP WITH TIME ZONE EXTERNAL**

TIMESTAMP WITH TIME ZONE EXTERNAL indicates a character string that represents a time stamp with a time zone indication. The default length is 33 bytes.
Valid lengths are 26 through 39 bytes. Time stamps with time zones have the following format:

\[ yyyy-mm-dd-hh.mm.ss.nnnnnn\pm th:tm \]

The \( nnnnnn \) microseconds value is optional. If not specified, it defaults to 000000. If the \( nnnnnn \) value has trailing zeroes, you can omit them.

You can omit leading zeroes from the month (\( mm \)), day (\( dd \)), and hour (\( hh \)) subfields.

**ROWID**

ROWID indicates a hexadecimal number that represents a DB2 internal row ID.

**CLOB**

CLOB indicates a character-based large object (LOB).

**BLOB**

BLOB indicates a binary-value-based LOB.

**DBCLOB**

DBCLOB indicates a double-byte character-based LOB.

**Data type considerations**

The following considerations apply to data types and their specifications in NGT Load.

**Data type conversions**

NGT Load enables you to perform flexible and extensive data type conversions. The following tables describe valid and invalid data conversions. An X denotes a valid conversion; an O denotes an invalid conversion.

**Table 6: Allowable data type conversions for numeric output columns**

<table>
<thead>
<tr>
<th>Inputs</th>
<th>SMALLINT</th>
<th>BIGINT</th>
<th>INTEGER</th>
<th>DECIMAL</th>
<th>FLOAT</th>
<th>DECFLOAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMALLINT</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
### Table 7: Allowable data type conversions for character and graphic output columns

<table>
<thead>
<tr>
<th>Inputs</th>
<th>CHAR</th>
<th>VARCHAR</th>
<th>LONG VARCHAR</th>
<th>GRAPHIC</th>
<th>VARGRAPHIC</th>
<th>LONG VARGRAPHIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMALLINT</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>BIGINT</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>INTEGER</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>INTEGER EXTERNAL</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DECIMAL PACKED</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>FLOAT</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>CHAR</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>GRAPHIC</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>Inputs</td>
<td>Output</td>
<td>CHAR</td>
<td>VARCHAR</td>
<td>LONG 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>DECIMAL EXTERNAL</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DECFLOAT</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
</tbody>
</table>

**Table 8: Allowable data type conversions for date/time output columns**

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Output</th>
<th>DATE</th>
<th>TIME</th>
<th>TIMESTAMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMALLINT</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>INTEGER</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>INTEGER EXTERNAL</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DECIMAL PACKED</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DECIMAL ZONED</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DECIMAL EXTERNAL</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>FLOAT EXTERNAL</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>DECFLOAT</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
</tbody>
</table>
Data type limitations

NGT Load has the following data type limitations:

- NGT Load supports large objects (LOBs) only if you specify SHRLEVEL CHANGE.

  **Note**
  
The LOB must be contained within a 32-KB input record.

- NGT Load does not support the following data types:
  - BINARY
  - VARBINARY
  - XML

Unsupported data type functions and specifications

NGT Load does not support the following data type functions and specifications:

- Rounding for DECFLOAT data
- **TRUNC** or **STRIP**
  
  TRUNC and STRIP are used in WHERE/WHEN criteria selection. NGT Load performs similar functions using a generalized data manipulation facility implemented under the VALUE keyword.

LOAD syntax that is no longer valid

The following LOAD options that were valid in the BMC LOADPLUS product are not valid for use with NGT Load. The following table indicates whether NGT Load ignores the option or terminates.

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DATE</td>
</tr>
<tr>
<td>GRAPHIC</td>
<td>O</td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td>O</td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td>X</td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td>O</td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td>X</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DATE</td>
</tr>
<tr>
<td>GRAPHIC</td>
<td>O</td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td>O</td>
</tr>
<tr>
<td>DATE EXTERNAL</td>
<td>X</td>
</tr>
<tr>
<td>TIME EXTERNAL</td>
<td>O</td>
</tr>
<tr>
<td>TIMESTAMP EXTERNAL</td>
<td>X</td>
</tr>
<tr>
<td>Invalid option</td>
<td>NGT Load action</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ANALYZE</td>
<td>Ignores ANALYZE SCAN and ANALYZE SAMPLE</td>
</tr>
<tr>
<td></td>
<td>Terminates on ANALYZE PAUSE and ANALYZE ONLY</td>
</tr>
<tr>
<td>AVAILPAGEPCT</td>
<td>Ignores</td>
</tr>
<tr>
<td>BMCSTATS</td>
<td>Ignores BMCSTATS NO</td>
</tr>
<tr>
<td></td>
<td>Terminates on BMCSTATS YES</td>
</tr>
<tr>
<td>CENTURY</td>
<td>Terminates</td>
</tr>
<tr>
<td>COPY</td>
<td>Ignores COPY YES INLINE</td>
</tr>
<tr>
<td></td>
<td>Ignores COPY NO COPYPEND</td>
</tr>
<tr>
<td></td>
<td>Terminates on COPY YES REGISTER NONE and COPY YES REGISTER  <em>ddname</em></td>
</tr>
<tr>
<td></td>
<td>(Honors all other values)</td>
</tr>
<tr>
<td>COPYLVL</td>
<td>Ignores</td>
</tr>
<tr>
<td>DDTYPE and all of its suboptions</td>
<td>Terminates</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> You can use the OUTPUT command to allocate discard and copy data sets. For more information, see “OUTPUT command” on page 31.</td>
</tr>
<tr>
<td>DECFLOAT_ROUNDMODE</td>
<td>Terminates</td>
</tr>
<tr>
<td>DELETEFILES SYSDISC</td>
<td>Ignores</td>
</tr>
<tr>
<td>DISCARDS IGNORE and DISCARDS REPORT</td>
<td>Ignores</td>
</tr>
<tr>
<td>DRAIN_WAIT</td>
<td>Terminates on DRAIN_WAIT NONE, DRAIN_WAIT SQL, and DRAIN_WAIT UTIL</td>
</tr>
<tr>
<td></td>
<td>(Honors DRAIN_WAIT <em>integer</em>)</td>
</tr>
<tr>
<td>DSNUEXIT</td>
<td>Ignores</td>
</tr>
<tr>
<td>DSNUTILB</td>
<td>Ignores</td>
</tr>
<tr>
<td>ENFORCE</td>
<td>Terminates on ENFORCE CHECK CONSTRAINTS, ENFORCE RI, and ENFORCE...INFORI</td>
</tr>
<tr>
<td></td>
<td>(Honors all other values)</td>
</tr>
<tr>
<td>ENUMROWS</td>
<td>Ignores</td>
</tr>
<tr>
<td>ERRDDN</td>
<td>Ignores</td>
</tr>
<tr>
<td>EXCHANGE</td>
<td>Terminates</td>
</tr>
<tr>
<td>FASTSWITCH</td>
<td>Ignores</td>
</tr>
<tr>
<td>FORCE</td>
<td>Ignores FORCE NONE</td>
</tr>
<tr>
<td></td>
<td>Terminates on all other FORCE keywords</td>
</tr>
<tr>
<td>FORMAT</td>
<td>Terminates on all keywords except FORMAT UNLOAD</td>
</tr>
<tr>
<td>Invalid option</td>
<td>NGT Load action</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>IDCACHE</td>
<td>Ignores</td>
</tr>
<tr>
<td>IDCDDN</td>
<td>Terminates</td>
</tr>
<tr>
<td>IDERROR</td>
<td>Terminates</td>
</tr>
<tr>
<td>IFDISCARDLIMIT</td>
<td>Terminates</td>
</tr>
<tr>
<td>IFDISCARDS</td>
<td>Terminates</td>
</tr>
<tr>
<td>IFZEROROWS</td>
<td>Terminates</td>
</tr>
<tr>
<td>IMPLICIT_TZ</td>
<td>Terminates</td>
</tr>
<tr>
<td>INDEX</td>
<td>Ignores INDEX BUILD</td>
</tr>
<tr>
<td></td>
<td>Terminates on INDEX UPDATE</td>
</tr>
<tr>
<td>INDSN</td>
<td>Terminates</td>
</tr>
<tr>
<td>INTO TABLE <em>tableName</em> REPLACE</td>
<td>Terminates</td>
</tr>
<tr>
<td></td>
<td>(Honors other INTO options)</td>
</tr>
<tr>
<td>KEEPDICTIONARY</td>
<td>Ignores</td>
</tr>
<tr>
<td>LOADDN</td>
<td>Ignores</td>
</tr>
<tr>
<td>LOADLOBDATA</td>
<td>Ignores</td>
</tr>
<tr>
<td>LOBAVGPC</td>
<td>Ignores</td>
</tr>
<tr>
<td>LOG</td>
<td>Terminates on LOG NO with RESUME YES</td>
</tr>
<tr>
<td></td>
<td>SHRLEVEL CHANGE</td>
</tr>
<tr>
<td></td>
<td>Ignores all other keywords</td>
</tr>
<tr>
<td>LONGNAME_TRUNC</td>
<td>Ignores</td>
</tr>
<tr>
<td>MAPDDN</td>
<td>Ignores</td>
</tr>
<tr>
<td>MAXSORTMEMORY</td>
<td>Ignores</td>
</tr>
<tr>
<td>MAXSORTS</td>
<td>Ignores</td>
</tr>
<tr>
<td>MAXTAPE</td>
<td>Ignores</td>
</tr>
<tr>
<td>MINSORTMEMORY</td>
<td>Ignores</td>
</tr>
<tr>
<td>NLPCTFREE</td>
<td>Ignores</td>
</tr>
<tr>
<td>NOSUBS</td>
<td>Terminates</td>
</tr>
<tr>
<td>ORDER</td>
<td>Ignores ORDER NO and ORDER PRESORTED</td>
</tr>
<tr>
<td></td>
<td>Terminates on ORDER YES ASSOCIATE</td>
</tr>
<tr>
<td></td>
<td>(Honors ORDER YES without ASSOCIATE)</td>
</tr>
<tr>
<td>ORIGINAL_DISP</td>
<td>Ignores</td>
</tr>
<tr>
<td>PERIODOVERRIDE</td>
<td>Ignores</td>
</tr>
<tr>
<td>Invalid option</td>
<td>NGT Load action</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>PREFORMAT at the partition level</td>
<td>Ignores (Honors PREFORMAT at the table space level)</td>
</tr>
<tr>
<td>PRELOAD</td>
<td>Ignores PRELOAD LOAD and PRELOAD CONTINUE Terminates on PRELOAD PAUSE and PRELOAD ANALYZE</td>
</tr>
<tr>
<td>RETRY</td>
<td>Ignores</td>
</tr>
<tr>
<td>RETRY_DELAY</td>
<td>Ignores</td>
</tr>
<tr>
<td>SHORTMEMORY</td>
<td>Ignores</td>
</tr>
<tr>
<td>SKIPIX</td>
<td>Ignores</td>
</tr>
<tr>
<td>SORTKEYS</td>
<td>Ignores</td>
</tr>
<tr>
<td>SQLAPPLY and most of its suboptions</td>
<td>Ignores (Honors APCOMMIT)</td>
</tr>
<tr>
<td>SWITCHTIME</td>
<td>Terminates</td>
</tr>
<tr>
<td>SYNC</td>
<td>Ignores</td>
</tr>
<tr>
<td>TOTALPAGEPCT</td>
<td>Ignores</td>
</tr>
<tr>
<td>TRANSIDOVERRIDE</td>
<td>Terminates</td>
</tr>
<tr>
<td>TSSAMPLEPCT</td>
<td>Ignores</td>
</tr>
<tr>
<td>UNIQUECHECK</td>
<td>Ignores</td>
</tr>
<tr>
<td>UNIQUEINTO</td>
<td>Ignores</td>
</tr>
<tr>
<td>UPDATEMAXA</td>
<td>Terminates</td>
</tr>
<tr>
<td>WORKDDN</td>
<td>Ignores</td>
</tr>
<tr>
<td>XBMID</td>
<td>Ignores</td>
</tr>
<tr>
<td>XMLAVGSIZE</td>
<td>Ignores</td>
</tr>
<tr>
<td>ZIIP</td>
<td>Ignores</td>
</tr>
</tbody>
</table>
Parameters for NGT Load

This chapter describes NGT Load parameters. Unless indicated otherwise, you include these parameters in the LODPARMS DD data set or in the NGT job stream.

Syntax rules for parameters

The following rules apply to the syntax for parameters for the Next Generation Technology (NGT) products:

- All parameters must start with a plus symbol (+).

- Parameters can start anywhere, but must be contained within the first 72 character positions of the line. Columns 73-80 are ignored and can contain sequence numbers.

- You can code parameters over multiple lines. Continuation is automatic.

- You can code multiple parameters on the same line.

- You can code comments by enclosing the comment between /* (beginning) and */ (end). You can place comments anywhere, including in the middle of a word. The utility removes all comment strings before checking syntax.

  **Note**

  If you specify parameters by using a DD statement in your JCL, do not begin a comment with /* in column one. Placing /* in column one causes IBM MVS to generate a //SYSIN statement for any subsequent records in the data set unless you use DLM= to change the JCL comment delimiter.

- You can nest comments, with no limit to the number of nested levels that you can use. Ensure that each /* has a corresponding */.

- When specifying a series of values, you must separate them with commas. Blanks, wherever they are coded, have no significance. The utility removes all blanks before checking syntax.
- If the same parameter is specified multiple times within the same input data set, the latest specification prevails.
- Do not use the plus symbol inside parentheses for any parameter.

**+AFTERACC**

+AFTERACC specifies the ACCESS parameter of the DB2 -START command to use after the load is complete. This parameter applies to the table space and all indexes.

- **RW**
  - RW starts the spaces in read/write mode.

- **RO**
  - NGT Load ignores RO.

- **UT**
  - NGT Load ignores UT.

- **STOP**
  - NGT Load ignores STOP.

- **NO**
  - NO returns the table space to the state that it was in before the start of the job. You may also use NONE.

**Example**

+AFTERACC(RW)

NGT Load issues the -START ACCESS(RW) command after a successful table space load.
**+ALLO**

+ALLO controls the value that NGT Load uses for the primary allocation quantity for a new table space. Include this parameter in the RRGPARMX DD data set to control index data set allocations.

**USED**

*(default)* USED tells NGT Load to use the space amount from the existing data set as the primary quantity for the new data set. This value is obtained from the high-used RBA in the VSAM catalog.

**ALLO**

ALLO tells NGT Load to use the currently allocated space amount from the existing data set as the primary quantity for the new data set. The value is obtained from the high-allocated RBA in the VSAM catalog.

**DB2**

DB2 tells NGT Load to use primary quantity and secondary quantities from the DB2 catalog as the values for the new data set. If the object is not DB2-managed or if the primary quantity (PRIQTY) value is less than the high-used RBA for the data set, NGT Load uses the high-allocated value.

**KEEPDB2**

KEEPDB2 tells NGT Load to use PQTY values from the DB2 catalog for the new data set. This keyword differs from +ALLO (DB2) in that +ALLO(DB2) is a conditional switch, whereas the +ALLO(KEEPDB2) parameter forces NGT Load to use the DB2 values.

If the data set is user-managed, NGT Load uses the values in use for primary quantity as it is defined to AMS for the new data set.

If a value of -1 is set for any object in the catalog for either PRIQTY or SECQTY, NGT Load correctly keeps the DB2 primary and secondary allocations.

**MXIG**

MXIG is similar to the MVS JCL SPACE parameter. MXIG requests that space allocated to the data set be the smaller of the primary quantity needed or the space available on the volume. This parameter affects only primary space.
allocation. Secondary extents should be available when you use this parameter.

**REUSE**

REUSE tells NGT Load to write the newly loaded data set over the original data set. +ALLO(REUSE) can be honored only if the VSAM cluster has the REUSE attribute and the existing size is at least large enough for the data set to be loaded.

**Example**

```
+ALLO(REUSE)
```

NGT Load uses the currently allocated space to determine the primary allocation for the new data set.

**+COMMIT**

+COMMIT specifies the number of records to insert before performing a commit during a LOAD RESUME YES SHRLEVEL CHANGE job.

```
+COMMIT { -1 integer }
```

The APCOMMIT syntax option overrides this parameter. For more information, see “APCOMMIT” on page 53.

-1

(default) This value tells NGT Load to perform a commit after all rows have been inserted.

*integer*

Specify an integer to tell NGT Load the maximum number of rows to insert before performing a commit.

**Example**

```
+COMMIT(10)
```

NGT Load performs a commit after it inserts every tenth row.
+DISCMSGS

+DISCMSGS indicates the maximum number of discard messages that you want NGT Load to print per partition. When this value is exceeded, NGT Load suppresses additional discard messages. If you are loading many partitions, consider a low value to prevent the printing of millions of redundant messages.

0

(default) 0 tells NGT Load to stop issuing discard messages after 50 messages.

integer

Any other integer indicates the maximum number of messages that you want NGT Load to print per partition. This value can be a number from 1 through 10,000.

Example

+DISCMSGS(1000)

NGT Load prints discard messages for the first 1,000 records discarded and suppresses the printing of additional discard messages.

+FILESZ

+FILESZ specifies the number of thousands of records in the input file.

integer

NGT Load uses this parameter to ensure the allocation of work files of the appropriate size. You must provide this value only if you are loading more than 200 million records. The default sort allocations handle most cases below 200 million records.

Specify an integer as an estimate of the number of records, in thousands, that NGT Load will load.
Example

+FILESZ(500000)

NGT Load allocates work space for 500 million records.

+PAD

+PAD tells NGT Load, when you specify FORMAT(UNLOAD), whether the VARCHAR input values are padded.

WARNING

Ensure that the +PAD option correctly corresponds to the input data. Otherwise, NGT Load loads bad data and DB2 might abend.

YES

(default) YES tells NGT Load that the VARCHAR input values are fully padded.

Note

The length that precedes the data in the VARCHAR field does not include the padding characters.

NO

NO tells NGT Load that the VARCHAR input values are not padded.

Example

If the loading of a column IDNUM is defined as VARCHAR(6), the parameter +PAD(YES) forces NGT Load to load the IDNUM column with trailing blanks, as in the following example. (In this example, each ^ signifies a blank space.)

<table>
<thead>
<tr>
<th>IDNUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC139</td>
</tr>
<tr>
<td>EK?8^^</td>
</tr>
<tr>
<td>R29^^^</td>
</tr>
</tbody>
</table>

The parameter +PAD(NO) loads the IDNUM column without trailing blanks, as in the following example:

<table>
<thead>
<tr>
<th>IDNUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>------</td>
</tr>
</tbody>
</table>
+RUNSTATS

+RUNSTATS activates statistics gathering and updates the DB2 catalog.

\[
+\text{RUNSTATS} \rightarrow \begin{cases} 
\text{NO} \\
\text{YES} 
\end{cases}
\]

The UPDATEDB2STATS option overrides the +RUNSTATS parameter. For more information, see “UPDATEDB2STATS” on page 50.

NO

NO tells NGT Load to not update the statistics.

YES

YES tells NGT Load to update the statistics according to the STTPARMS. The STTPARMS DD is the location in which the parameters for BMC Next Generation Technology Stats for DB2 for z/OS (NGT Stats) are specified. For more information, see the BMC Next Generation Technology Stats for DB2 for z/OS Reference Manual.

Alternatively, you can specify ALL instead of YES.

Restrictions

The +RUNSTATS parameter is subject to the following restrictions:

- NGT Load does not gather RUNSTATS during LOAD RESUME YES processing.

- This parameter computes aggregate statistics using catalog values for partitions that are not processed. To ensure that meaningful values are computed when processing a subset of a partitioned table space, you must run NGT Stats on the entire table space when, or immediately after, the table space has loaded to collect statistics for all partitions.

- NGT Load skips non-indexed tables.

- The statistics that NGT Load gathers update SYSCOLUMNS for the first column only of each index. Run NGT Stats to obtain statistics for all columns of all indexes.
Additional considerations

The following additional considerations apply to the +RUNSTATS parameter:

- NGT Load updates column statistics for every column of every index.

- You can also have NGT Load collect frequency distributed statistics can also be collected. The NGT Stats parameters (from the STTPARMS DD) control frequency statistics collection. If the STTPARMS DD does not supply these parameters for the load job, NGT Load obtains them from the NGT configuration.

- For partitioned table spaces, NGT Load does not update table statistics unless there is an index on the partitioning columns or a DPSI defined.

- For a list of which DB2 catalog statistics you can update, see the BMC Next Generation Technology Stats for DB2 for z/OS Reference Manual.

Example

```
+RUNSTATS(YES)
```

NGT Load gathers statistics and updates the DB2 catalog with the new values.

+SETCHECKPEND

+SETCHECKPEND enables you to tell NGT Load whether to set CHECK pending status on table spaces that are involved in RI (referential integrity) constraints.

```
+SETCHECKPEND(YES)
```

The CHECKPEND option overrides the value of this parameter. For more information, see “CHECKPEND” on page 50.

- YES

  (default) YES tells NGT Load to set the CHECK pending flag for the table space.

- NO

  NO tells NGT Load to not set the CHECK pending flag for the table space or any of its children.
Example

+SETCHECKPEND(YES)

NGT Load sets the CHECK pending flag.

+SORT

+SORT enables you to tell NGT Load whether to sort data in clustering index order.

If you know that the data has already been sorted, specify NO to prevent the sort. Specify YES when your data is not in order and you want a LOAD REPLACE to result in a clustered table space that does not need to be reorganized.

NO

(default) NO tells NGT Load to not sort the input data before loading.

The ORDER YES option overrides +SORT(NO). For more information, see “ORDER YES” on page 48.

YES

YES tells NGT Load to sort the input data in clustering index order before loading.

Example

+SORT(YES)

NGT Load does not sort the input data in clustering index order before loading.

+WRITETO

+WRITETO enables you to tell NGT Load whether to write to source or shadow data sets during the load.
This parameter is relevant only for LOAD REPLACE SHRLEVEL NONE (where SHRLEVEL NONE is either specified or implicit by default). LOAD RESUME YES always uses a non-destructive load technique. NGT Load ignores this parameter when you specify SHRLEVEL CHANGE or SHRLEVEL REFERENCE.

Use this parameter when you need to change all LOAD REPLACE jobs (that do not already use SHRLEVEL CHANGE or SHRLEVEL REFERENCE) to function as if you had specified SHRLEVEL REFERENCE.

A  
(default) A tells NGT Load to write to the source data set and not create shadow data sets.

B  
B tells NGT Load to create and write to shadow data sets. At the end of the load, rename or fastswitch processing takes place.

**Example**

+WRITETO(B)

NGT Load completes the following process:

1. Writes all data pages to shadow data sets.
2. Performs fastswitch processing (if +FASTSWITCH is YES) or rename processing.
3. Deletes the original data sets.
Syntax examples

This chapter describes the syntax for various tasks that you can perform using NGT Load.

For the examples in this chapter, assume that the following objects exist:

```sql
-- NON-PARTITIONED TABLESPACE
CREATE TABLE NGT.TABLE1
(ID CHAR(3) NOT NULL ,
 MESSAGE CHAR(34) ,
 LANGUAGE CHAR(10) NOT NULL )
IN DATABASE DSNDB04 ;

-- PARTITIONED TABLESPACE (3 PARTS)
CREATE TABLESPACE PARTTS IN DSNDB04 NUMPARTS 3;
CREATE TABLE NGT.PART_TABLE1
(ID CHAR(3) NOT NULL ,
 MESSAGE CHAR(34) ,
 LANGUAGE CHAR(10) NOT NULL WITH DEFAULT )
IN DSNDB04.PARTTS;
CREATE INDEX NGT.INDEX1 ON NGT.TABLE1
(ID ASC) CLUSTER
(PART 1 VALUES ('003') ,
 PART 2 VALUES ('006') ,
 PART 3 VALUES ('010') ) ;
```

For the examples in this chapter, assume that NGT Load uses the following input file:

```
001THANK YOU FOR PURCHASING NGT                ENGLISH
002THANK YOU FOR PURCHASING NGT                ITALIAN
003                                            DUTCH
004THANK YOU FOR PURCHASING NGT                GERMAN
005THANK YOU FOR PURCHASING NGT                FRENCH
006THANK YOU FOR PURCHASING NGT                SPANISH
007NGT SEIHIN WO OKAITE ARIGATOU GOZAIMASITA  JAPANESE
008SHUKRAN LE SHEERAH AKUM NGT                 ARABIC
```

The scale shown above the data (+----+----+) provides a visual reference for the relative column positions of the data. Assume that the sample data is in a sequential file named USER.DATA.
Replace data without allowing concurrent access

The following example illustrates a typical LOAD REPLACE in which you load new data into a table space that has been deleted and redefined. The job defaults to SHRLEVEL NONE, causing NGT Load to stop the table space before loading the data.

For this example, make the assumptions described in “Syntax examples” on page 85.

```
//SYSREC    DD DISP=SHR,DSN=USER.DATA
//SYSIN     DD *
LOAD DATA REPLACE INDDN SYSREC
INTO TABLE NGT.TABLE1
(ID    POSITION(2),
 MESSAGE    POSITION(*)  CHAR(34) NULLIF(1:1) = '?',
 LANGUAGE    POSITION(*)  CHAR(10) DEFAULTIF(1:1) = '?')
```

Replace data while allowing concurrent read access

This example illustrates the ability to run a LOAD REPLACE job while leaving the table space in read-only (RO) status.

This example uses the SHRLEVEL REFERENCE option to tell NGT Load to leave the original table space in RO status during the load and create shadow data sets. NGT Load loads the new data to these shadow data sets, and then stops the table space to perform rename or fastswitch processing.

The benefit of this job is data availability; the cost is the additional DASD required for the shadow copies during the load process.

For this example, make the assumptions described in Syntax examples on page 85.

```
//LODPARMS  DD *
//SYSREC    DD DISP=SHR,DSN=USER.DATA
//SYSIN     DD *
LOAD DATA REPLACE SHRLEVEL REFERENCE INDDN SYSREC
INTO TABLE NGT.TABLE1
(ID    POSITION(2),
 MESSAGE    POSITION(*)  CHAR(34) NULLIF(1:1) = '?',
 LANGUAGE    POSITION(*)  CHAR(10) DEFAULTIF(1:1) = '?')
```

Note

Alternatively, you can use the +WRITETO(B) parameter instead of SHRLEVEL REFERENCE.
Add data to an existing table space

The following example illustrates a typical LOAD RESUME YES in which NGT Load appends the provided data to the existing data in the table. During the load, NGT Load stops the table space.

For this example, make the assumptions described in Syntax examples on page 85.

```
//SYSREC DD DISP=SHR,DSN=USER.DATA
//SYSIN DD *
LOAD DATA RESUME YES INDDN SYSREC
  INTO TABLE NGT.TABLE1
  (ID    POSITION(2),
   MESSAGE    POSITION(*)  CHAR(34) NULLIF(1:1) = '?',
   LANGUAGE    POSITION(*)  CHAR(10) DEFAULTIF(1:1) = '?')
```

Replace one partition while allowing concurrent READ access

The following example illustrates how to replace one partition of the table space and leave the existing data in the other partitions. In this example, the partition that is being loaded is in RO status and the remaining partitions, as well as the indexes on the table space, are in RW status.

If the data for other partitions is in the input file, NGT Load discards that data because it does not belong in the partition being loaded.

For this example, make the assumptions described in Syntax examples on page 85.

```
//LODPARMS DD *
+WRITETO(B)
//SYSREC DD DISP=SHR,DSN=USER.DATA
//SYSIN DD *
LOAD DATA INDDN SYSREC
  INTO TABLE NGT.TABLE1 PART 1 REPLACE
  (ID    POSITION(2),
   MESSAGE    POSITION(*)  CHAR(34) NULLIF(1:1) = '?',
   LANGUAGE    POSITION(*)  CHAR(10) DEFAULTIF(1:1) = '?')
```

Replace one partition without allowing concurrent access

The following example illustrates how to replace one partition of the table space and leave the existing data in the other partitions. In this example, the partition that is
being loaded is stopped and the remaining partitions, as well as the indexes on the
table space, are in RW status.

If the data for other partitions is in the input file, NGT Load discards that data
because it does not belong in the partition being loaded.

For this example, make the assumptions described in Syntax examples on page 85.

```plaintext
//SYSREC    DD DISP=SHR, DSN=USER.DATA
//SYSIN     DD *
LOAD DATA INDDN SYSREC
INTO TABLE NGT.TABLE1 PART 1 REPLACE
  (ID    POSITION(2),
   MESSAGE    POSITION(*)  CHAR(34) NULLIF(1:1) = '?',
   LANGUAGE    POSITION(*)  CHAR(10) DEFAULTIF(1:1) = '?')
```

Replace some partitions, add data to some partitions, and leave remaining partitions available

The following example illustrates replacing the data in some partitions, adding to
existing data in other partitions, and leaving the remaining partitions in RW status.

This particular example replaces partitions 2, 4, and 6, and adds data to partitions 1,
3, and 5. During this load, NGT Load stops partitions 1 through 6. All other
partitions and all indexes remain in RW status.

For this example, make the assumptions described in Syntax examples on page 85.

```plaintext
//SYSREC    DD DISP=SHR, DSN=USER.DATA
//SYSIN     DD *
LOAD DATA RESUME YES INDDN SYSREC
INTO TABLE NGT.TABLE1 PARTS 2,4,6 REPLACE
  (fieldSpecification)
INTO TABLE NGT.TABLE1 PARTS 1,3,5 RESUME YES
  (fieldSpecification)
```

Add data to an existing table space while allowing read/write access

The following example, illustrates adding data to existing data while leaving the
table space in RW status. NGT Load uses INSERT statements to insert records from
the input file into the table.
For this example, make the assumptions described in Syntax examples on page 85.

```plaintext
//SYSREC    DD DISP=SHR,DSN=USER.DATA
//SYSIN     DD *
LOAD DATA RESUME YES SHRLEVEL CHANGE INDDN SYSREC
INTO TABLE NGT.TABLE1
(fieldSpecification)
```

**Remove all data from a table space**

The following example illustrates how to use NGT Load remove all data from a table space.

Specifying REPLACE with an input data set of DD DUMMY empties the table space and loads no data.

For this example, make the assumptions described in “Syntax examples” on page 85.

```plaintext
//SYSREC    DD DUMMY
//SYSIN     DD *
LOAD DATA REPLACE INDDN SYSREC
INTO TABLE NGT.TABLE1
(fieldSpecification)
```

**Load delimited data**

The following examples illustrate how to use NGT Load to load delimited data.

This example uses the same LOAD statement syntax as the example for emptying a table space, but loads delimited data instead of a dummy data set.

```plaintext
//SYSREC    DD DISP=SHR,DSN=USER.DATA
//SYSIN     DD *
LOAD DATA RESUME NO REPLACE INDDN SYSREC
FORMAT DELIMITED
INTO TABLE NGT.TABLE1
```
Note
In the following examples:

- Make the assumptions described in Syntax examples on page 85.
- Leading blanks that are not delimited create white space and are not part of the target field.
- Code omitted fields as two consecutive column delimiters, as shown in the second field of the second input record of the second example.
- NGT Load loads null or default values for omitted fields.
- The characteristics of fields coded as two consecutive character delimiters depends on the target field, as described in the following table:

### Table 9: Consecutive character delimiters

<table>
<thead>
<tr>
<th>Target field</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Varying</td>
<td>0 length</td>
</tr>
<tr>
<td>Fixed character</td>
<td>Blanks</td>
</tr>
<tr>
<td>Numeric, date, or time</td>
<td>Generates an error</td>
</tr>
</tbody>
</table>

### Data format example 1 – Delimited file with delimited character strings

This example illustrates a delimited input file that uses delimited character strings. In this example, the column delimiter is a comma (,). Because the character strings contain the column delimiter character, you must delimit them with character string delimiters. The character string delimiter is a double quotation mark (").

```
"009","Delimited fields","English"
"010","Delimited fields","Spanish"
```

### Data format example 2 – Delimited file with non-delimited character strings

This example illustrates a delimited file that uses non-delimited character strings. In this example, the column delimiter is a semicolon (;). Because the character strings do not contain the column delimiter character, they do not need to be delimit with character string delimiters.

```
009; Delimited fields;English
010;;Spanish
```
Use IGNOREFIELDS with the VALUE option to convert data

This topic contains examples that illustrate several uses of the IGNOREFIELDS option. Alternatively, you can use the SKIPFIELDS option to achieve the same results.

In the following examples, you tell NGT Load to ignore a field in the input data by giving the field a name that is not a column of the table. In these examples, the name is TEMP01. NGT Load uses that field in a table column value clause, along with SQL functions, to convert the field to the desired value for input in the load.

**Note**

These examples do not use the same objects as the other examples in this chapter.

**TIMESTAMP to CHAR**

This example tells NGT Load to ignore the TIMESTAMP field, convert the data to character data, and load it into a CHAR column. COL01 is defined in the table as CHAR(40).

```sql
//SYSREC DD *
2000-07-31-10.30.15.610208
//SYSIN DD *
LOAD DATA INDDN(SYSREC)
RESUME YES
INTO TABLE NGT.TABLE
IGNOREFIELDS YES
(TEMP01 POSITION(1:28) CHAR,
COL01 VALUE(CHAR(DATE(TIMESTAMP(TEMP01))))
)
```

**SUBSTR**

This example demonstrates the use of SUBSTR.

```sql
//SYSREC DD *
1234567
//SYSIN DD *
LOAD DATA INDDN(SYSREC)
RESUME YES
INTO TABLE NGT.TABLE
IGNOREFIELDS YES
(TEMP01 POSITION(1:5) CHAR,
COL01 VALUE(SUBSTR(TEMP01,1,4))
)
```
**TIMESTAMP to TIME**

This example tells NGT Load to ignore the TIMESTAMP field and load the data into a TIME column. COL02 is defined in the table as TIME.

```plaintext
//SYSREC    DD *
12009-08-03-09.02.15.610208
//SYSIN     DD *
LOAD DATA INDDN(SYSREC)
RESUME YES
INTO TABLE  NGT.TABLE
IGNOREFIELDS YES
(COL01  POSITION(1:1)  CHAR,
 TEMPO2 POSITION(2:29) CHAR,
 COL02  VALUE(TIME(TIMESTAMP(TEMPO2)))
)
```

**TIMESTAMP to INTEGER**

This example tells NGT Load to ignore the TIMESTAMP field, convert the data to numeric data, and load it into an INTEGER column. COL02 is defined in the table as INTEGER.

```plaintext
//SYSREC    DD *
12009-08-03-09.04.17.610208
//SYSIN     DD *
LOAD DATA INDDN(SYSREC)
RESUME YES
INTO TABLE  NGT.TABLE
IGNOREFIELDS YES
(COL01  POSITION(1:1)  CHAR,
 TEMPO2 POSITION(2:29) CHAR,
 COL02  VALUE(DAYS(TIMESTAMP(TEMPO2)))
)
```

**FLOAT to REAL**

This example tells NGT Load to ignore the FLOAT field and load the data into a REAL field. COL02 is defined in the table as REAL.

```plaintext
//SYSREC    DD *
+0.234000038E+01
//SYSIN     DD *
LOAD DATA INDDN(SYSREC)
RESUME YES
INTO TABLE  NGT.TABLE
IGNOREFIELDS YES
(COL01  POSITION(2:2)  CHAR,
 TEMPO2 POSITION(1:17) CHAR,
 COL02  VALUE(REAL(TEMPO2)))
)
```

**FLOAT to DECIMAL**

This example tells NGT Load to ignore the FLOAT field, convert the data to decimal format, and load it into a DECIMAL field. COL02 is defined in the table as DECIMAL(7,4).

```plaintext
//SYSREC    DD *
+0.345600038E+01
```
FLOAT to DECIMAL to CHAR

This example tells NGT Load to ignore the FLOAT field, convert the data first to DECIMAL(7,4) then to character data, and load it into a CHAR column. COL01 is defined in the table as CHAR.
Use IGNOREFIELDS with the VALUE option to convert data
Recommended automation control points for NGT Load

NGT automation control points, which are integrated into all NGT utilities, can enhance and expand processing options. For more information about automation control points, see the *BMC Next Generation Technology Automation Reference Manual*.

NGTAUTO DD statements trigger automation control point processing, as in the following example:

```
//NGTAUTO DD DISP=SHR, DSN=NGT.AUTO.CTRL.POINTS
```

The automation control point data set contains one member with each of the automation control points selected for use.

---

**Note**

If you do not include an NGTAUTO DD statement in your JCL, NGT Load uses the values specified in the +NGTAUTO1 or +NGTAUTO2 parameters (either specified in the configuration or overridden in the UTLPARMS DD).

BMC recommends that you use the following NGT automation control points with NGT Load. However, you can also use other automation control points.

<table>
<thead>
<tr>
<th>Automation control point</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XSUTGLOB</td>
<td>Set global variables that multiple automation control points use</td>
</tr>
<tr>
<td>XSUT0000</td>
<td>Stop a job before it starts. The utility calls this automation control point before processing starts.</td>
</tr>
<tr>
<td>XSUTSYIN</td>
<td>Modify SYSIN statements. The utility calls this automation control point as it reads each SYSIN statement.</td>
</tr>
<tr>
<td>Automation control point</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| XSUTTERM                | Perform additional termination processing (for example, updating DB2 statistics or sending an audit report)  
The utility calls this automation control point after processing completes. |
| XSVRXERR                | Set an action to process on an error condition (for example, alerting a user to an error)  
The utility calls this automation control point if processing ends with an error condition. |
Index

@ROWNUM 59
+COMMIT parameter 78
+DISCMSGS parameter 79
+FILESIZ 23
+FILESZ parameter 79
+SORT 23

A

after access 20
ALLO 77
AMS 77
APCOMMIT option 53
ARCHIVE 43, 59
ASCII. See input data formats
authorizations 13
automation control point processing 95

B

BASIC 36
basic sequential access method. See BSAM
before access 20
BIGINT 60
BINARY data type 25
BRF 36
BSAM 35

C

catalog tables 18, 25
CCSID 43
CDCEXEC 27
CHANGE 38
CHAR 25
CHARACTER 60
CHARDEL 43
CHECK constraint clause 16
check pending 82
CHECK pending (CHKP) status
setting 50
CHECKPEND command option 50
CHKP (CHECK pending) status
setting 50
close tables 18, 25
COLDEL 43
copy data sets
registering 55
copy registration
specifying 55
COPYDDN 53
COPYDDN option 23
COPYDICTIONARY option 35

data manipulation 13
data sets, DB2
redefining 48
data type conversions 13
DATE 60
DATE EXTERNAL 62
DB2 77
optimizer, influencing 50
DECFLOAT 60, 62
DECFLOAT EXTERNAL 62
DECIMAL 60
DECIMAL EXTERNAL 62
DECPT 43
DEFAULTIF option 62
DEFAULTIF(condition) 62
DELETEFILES option 49
deleting work files
command option 49
DELIMITED 43
delimited tokens 30
differences between LOADPLUS and NGT Load 25
discard data set 29
discard processing 13
DISCARDDN 17
DISCARDDN option 29, 46
DISCARDS option 46
DISORG 27
documentation information 9
drain
specifying the timeout value 52
DRAIN_WAIT option 52
DSNTIAUL 59
duplicate key 17
duplicate key errors 13
duplicate keys 20

E
EBCDIC. See input data formats
EBCDIC characters 62
encrypted tables 18, 25
ENFORCE 45
ERRDDN option 46
expression-based indexes 17
EXTENDED 36

F
field selection clause 59
field selection criteria 59
FLOAT 60
FLOAT EXTERNAL 62
FLOAT option 42
FORMAT 16, 43, 59, 80
FORMAT ARCHIVE 59
FORMAT UNLOAD 59

G
GENERATED ALWAYS 13
GENERATED ALWAYS identity columns, overriding 51
GENERATED BY DEFAULT 13
global parameters 27

H
hash table space 25
hash-organized tables 18
high-used RBA 77

I
identity columns 13, 89
loading from an input file 51
overriding GENERATED ALWAYS columns 51
IDENTITYOVERRIDE option 51
IEEE 42
IGNOREFIELDS 57
IGNOREFIELDS option 91
image copy 53
INDDN 35
INDDN option 28
INDDN specifications 25
Index Term 53, 54, 85
indexes
redefining 48
indexes on expression 13, 17
input data formats
ASCII 42
EBCDIC 42
input file 28
INTEGER 60
integrity checking 13
INTO TABLE clauses 56

K
KEEPDB2 77
KEEPDICTIONARY 25
KEEPDICTIONARY option 41
L
large objects. See LOBs
LB (Local Backup) 53
LimitKey error 17
LOAD DATA 27
LOAD DATA statement 16
LOAD RESUME SHRLEVEL NONE 23
loading multiple tables 13
LOADPLUS 25
LOADPLUS syntax 71
LOBs 25
Local Backup. See LB
local copy 53
Local Primary. See LP
LODPARMS 27
LODPARMS DD 75
LOG option 42
LOG YES 25
long object names 30
LP (Local Primary) 53

O
OBID 16, 41
object names, long 30
ORDER option
description 48
OUTPUT command 28, 29, 31
overriding OBIDs 13

P
packed decimal numbers 62
PART 57
partition-by-growth table spaces 18
partitioned table spaces 43
partitioned tables 57
PQTY 77
PREFORMAT option 38
publications, related 9

Q
QUICKEXIT 19, 20

R
RB (Remote Backup) 53
RECFM 27
RECOVERY utility 19
recovery, LOAD RESUME YES 13
RECOVERYDDN 53
REDEFINE option
call 48
redefining data sets
REDEFINE option 48
referential integrity. See RI
REGISTER option of COPY YES option
description 55
registering copies
specifying 55
related publications 9
Remote Backup. See RB
Remote Primary. See RP
REPLACE 36
RESTART 20
RESUME 36

N
names of objects, long 30
NGTAUTO 95
NGTTAPE 53
NONE 36
nonpartitioned indexes 13
NOPAD 59
NOT NULL 59
NULLIF option 62
NULLIF(condition) 62
RESUME NO 25
REUSE 77
REUSE option 49
RI 82
rounding options 25
ROWFORMAT 36
ROWID 21
ROWID columns 13
RP (Remote Primary) 53
RRF 36
RRGPARMS 27
rules for command syntax 30
RUNSTATS 27

S
S390 42
security labels 13
SHRLEVEL 38
SHRLEVEL NONE 25
SKIPFIELDS 57
skipping fields 57
SMALLINT 60
SORTDEVT option 47
sorting data 13
specifying 48
sorting input data 48
SORTNUM option 48
SQL/DS data format 25
statistics
DB2 catalog 50
STOGROUP-defined data sets
redefining 48
STRIP 25, 57
SUBSTR keyword 91
switch phase 20
syntax
rules for specifying 30
SYSDISC 27
SYSDISC DD 17, 29
SYSOUT 27
SYSREC 27
SYSREC DD 28

T
table check constraints
CHECKPEND command option 50

table name 56
table spaces
redefining 48
TIME 60
TIME EXTERNAL 62
timeout
specifying 52
TIMESTAMP 60
TIMESTAMP EXTERNAL 62
TIMESTAMP WITH TIME ZONE 62
tokens in syntax, rules for 30
TRUNC 25

U
Unicode 42
Unicode VARGRAPHIC 25
UNLOAD 80
UPDATEDB2STATS option 50
updating statistics
DB2 catalog 50
USED 77
UTLPARMS 27

V
VALUE clause 57
VARBINARY data type 25
VARCHAR 25, 60, 80
VSAM catalog 77
VSAM cluster 77
VSAM data sets, redefining during the load 48

W
WHEN 59
WK00001 25
work data set 25
WORKDDN option 48

X
XCPYDYNM 53
XML data 25
XSUT0000 95
XSUTGLOB 95
XSUTSYIN 95
XSUTTERM 95
XSVRXERR 95

zoned format 62

Z

zIIP processing 13