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

**Address**  
BMC SOFTWARE INC  
2103 CITYWEST BLVD  
HOUSTON TX 77042-2827 USA

**Telephone**  
1 713 918 8800  
or  
1 800 841 2031

**Fax**  
1 713 918 8000

Outside United States and Canada

**Telephone**  
+01 713 918 8800

**Fax**  
+01 713 918 8000

© 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, FlashCopy, IBM, MVS, S/390, 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 UNPUBLISHED—RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES.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 ................................................................. 11
- Conventions ............................................................................. 12
- Syntax diagrams ..................................................................... 12

## Chapter 1  Overview of NGT Unload

## Chapter 2  Operational considerations
- NGT Unload requirements and restrictions ............................ 17
  - NGT Unload software requirements ........................................... 17
  - Required authorizations for NGT Unload ................................. 17
  - Status requirements for NGT Unload ........................................ 18
  - NGT Unload restrictions .............................................................. 18
- Sizing the work data set ............................................................ 18
- Selecting data to unload ............................................................. 18
  - Selecting columns ................................................................... 18
  - Selecting rows ......................................................................... 19
  - Partition filtering ...................................................................... 20
- Unloading from other sources .................................................... 24
- Unloading from dropped tables .................................................. 25
- Saving space when using the DSNTIAUL format ....................... 26
- Page encoding (UNLD command) ............................................... 27
- Managing the generated LOAD statement data set .................. 28
- Server restrictions ..................................................................... 29
- Differences between UNLOAD PLUS and NGT Unload ............... 30

## Chapter 3  Input for NGT Unload
- DD statements ........................................................................... 33
- NGT Unload syntax overview .................................................... 35
  - Command syntax rules for NGT utilities ................................... 36
  - Specifying object names in your unload syntax .......................... 37
- OUTPUT command ..................................................................... 37
- NGT Unload syntax diagrams .................................................... 37
- Alphabetical listing of NGT Unload options ............................... 46
- NGT Unload option descriptions—UNLOAD syntax .................. 49
  - UNLOAD .............................................................................. 50
<table>
<thead>
<tr>
<th>Option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNLOAD TABLESPACE</td>
<td>50</td>
</tr>
<tr>
<td>ASCII</td>
<td>51</td>
</tr>
<tr>
<td>AUTOTAG</td>
<td>51</td>
</tr>
<tr>
<td>BLANKPAD</td>
<td>52</td>
</tr>
<tr>
<td>BYPART</td>
<td>53</td>
</tr>
<tr>
<td>CCSID</td>
<td>53</td>
</tr>
<tr>
<td>CNTLCARDS</td>
<td>54</td>
</tr>
<tr>
<td>CNTLLDN</td>
<td>55</td>
</tr>
<tr>
<td>DATEFMT</td>
<td>55</td>
</tr>
<tr>
<td>DELETEFILES</td>
<td>56</td>
</tr>
<tr>
<td>DIRECT</td>
<td>57</td>
</tr>
<tr>
<td>DISCARDS</td>
<td>59</td>
</tr>
<tr>
<td>DRAIN_WAIT</td>
<td>60</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>61</td>
</tr>
<tr>
<td>EXCLUDE PART</td>
<td>62</td>
</tr>
<tr>
<td>FILL</td>
<td>62</td>
</tr>
<tr>
<td>FILTERPART</td>
<td>63</td>
</tr>
<tr>
<td>FIXEDVARCHAR</td>
<td>64</td>
</tr>
<tr>
<td>FORCE</td>
<td>66</td>
</tr>
<tr>
<td>FORMAT</td>
<td>66</td>
</tr>
<tr>
<td>INFILE</td>
<td>68</td>
</tr>
<tr>
<td>LIMIT</td>
<td>72</td>
</tr>
<tr>
<td>MINROWS</td>
<td>73</td>
</tr>
<tr>
<td>NOSUBS</td>
<td>73</td>
</tr>
<tr>
<td>NULLCHAR</td>
<td>73</td>
</tr>
<tr>
<td>NULLTYPE</td>
<td>74</td>
</tr>
<tr>
<td>ON FAILURE</td>
<td>75</td>
</tr>
<tr>
<td>ORDER</td>
<td>76</td>
</tr>
<tr>
<td>PART or LOGICAL PART</td>
<td>77</td>
</tr>
<tr>
<td>RETRY</td>
<td>78</td>
</tr>
<tr>
<td>RETRY_DELAY</td>
<td>78</td>
</tr>
<tr>
<td>SHRLEVEL</td>
<td>79</td>
</tr>
<tr>
<td>SPANNED</td>
<td>80</td>
</tr>
<tr>
<td>TIMEFMT</td>
<td>81</td>
</tr>
<tr>
<td>TSFMT</td>
<td>83</td>
</tr>
<tr>
<td>UNICODE</td>
<td>84</td>
</tr>
<tr>
<td>UNLOADADDN</td>
<td>85</td>
</tr>
<tr>
<td>ZONEDDECOVP</td>
<td>85</td>
</tr>
<tr>
<td>SELECT statement options</td>
<td>86</td>
</tr>
<tr>
<td>Data type keywords</td>
<td>99</td>
</tr>
<tr>
<td>Supported data type conversions</td>
<td>113</td>
</tr>
<tr>
<td>Topic</td>
<td>Page</td>
</tr>
<tr>
<td>---------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Additional data type considerations</td>
<td>121</td>
</tr>
<tr>
<td>Data translation</td>
<td>122</td>
</tr>
<tr>
<td>UNLOAD syntax that is no longer valid</td>
<td>128</td>
</tr>
<tr>
<td>NGT Unload option descriptions—UNLD syntax</td>
<td>129</td>
</tr>
<tr>
<td>APPEND</td>
<td>130</td>
</tr>
<tr>
<td>ASCII</td>
<td>130</td>
</tr>
<tr>
<td>BLANKPAD</td>
<td>130</td>
</tr>
<tr>
<td>BYPART</td>
<td>131</td>
</tr>
<tr>
<td>CCSID</td>
<td>131</td>
</tr>
<tr>
<td>COPYDDN</td>
<td>132</td>
</tr>
<tr>
<td>COPYDSN</td>
<td>132</td>
</tr>
<tr>
<td>DATA</td>
<td>133</td>
</tr>
<tr>
<td>DATEFMT</td>
<td>133</td>
</tr>
<tr>
<td>DIGITS</td>
<td>134</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>134</td>
</tr>
<tr>
<td>EXCLUDE PARTS</td>
<td>134</td>
</tr>
<tr>
<td>FORMAT</td>
<td>135</td>
</tr>
<tr>
<td>FROM [TABLE]</td>
<td>137</td>
</tr>
<tr>
<td>MAXERR</td>
<td>139</td>
</tr>
<tr>
<td>MAXLRECL</td>
<td>139</td>
</tr>
<tr>
<td>MAXROWS</td>
<td>139</td>
</tr>
<tr>
<td>MINROWS</td>
<td>140</td>
</tr>
<tr>
<td>MODE</td>
<td>140</td>
</tr>
<tr>
<td>NOFILTER</td>
<td>145</td>
</tr>
<tr>
<td>OBID</td>
<td>145</td>
</tr>
<tr>
<td>ORDER BY</td>
<td>145</td>
</tr>
<tr>
<td>OUTDDN</td>
<td>146</td>
</tr>
<tr>
<td>PAD or NOPAD</td>
<td>147</td>
</tr>
<tr>
<td>PARTS or LPARTS</td>
<td>147</td>
</tr>
<tr>
<td>ROWEXIT</td>
<td>148</td>
</tr>
<tr>
<td>SELECT</td>
<td>148</td>
</tr>
<tr>
<td>SHRLEVEL CHANGE</td>
<td>149</td>
</tr>
<tr>
<td>SPANNED YES</td>
<td>149</td>
</tr>
<tr>
<td>STMTDDN</td>
<td>150</td>
</tr>
<tr>
<td>SYSCOPY</td>
<td>151</td>
</tr>
<tr>
<td>TIMEFMT</td>
<td>151</td>
</tr>
<tr>
<td>TSFMT</td>
<td>152</td>
</tr>
<tr>
<td>UNICODE</td>
<td>153</td>
</tr>
<tr>
<td>WHERE</td>
<td>154</td>
</tr>
<tr>
<td>ZEROPAD</td>
<td>154</td>
</tr>
</tbody>
</table>
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.
Tip

If you prefer hardcopy documentation, you can order it from your BMC sales representative or from Support Central. Also, from Support Central you can subscribe to receive proactive e-mail alerts when BMC issues notices.

Conventions

This document uses the following special conventions:

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

- Variable text in path names, system messages, or syntax is displayed in italic text: `testsys/instance/fileName`

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

Syntax diagrams

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

The following guidelines provide additional information about syntax diagrams:

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

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

- An underlined item is a default option.

- If a diagram shows punctuation marks, parentheses, or similar symbols, you must enter them as part of the syntax.

- In general, IBM commands, keywords, clauses, and data types are displayed in uppercase letters. However, if an item can be shortened, the minimum required portion might be shown in uppercase letters, with the remainder in lowercase (for example, CANcel).

- The following conventions apply to variables in syntax diagrams:

  - Variables are typically displayed in lowercase letters and are always italicized.

  - If a variable is represented by two or more words, initial capitals distinguish the second and subsequent words (for example, databaseName).
Overview of NGT Unload

The BMC Next Generation Technology Unload for DB2 for z/OS utility unloads database objects to sequential data sets.
Operational considerations

This chapter provides information that you need to know to run the NGT Unload utility.

NGT Unload requirements and restrictions

This topic describes NGT Unload requirements and restrictions.

NGT Unload software requirements

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

Required authorizations for NGT Unload

Using NGT Unload requires the following authorizations:

- One of the following for the database:
  - SYSADM
  - DBADM
  - COPY

- One of the following for the tables:
  - Ownership
  - SELECT
  - SYSCTRL (catalog tables only)
Status requirements for NGT Unload

When running NGT Unload, the objects must be in the following statuses:

- The database must be in either RW status or UT,RO status. In addition, it cannot be in any restricted status.
- The table space must be in RW status and cannot be in any restricted status.

NGT Unload restrictions

NGT Unload does not process the following objects:

- Hash tables
- Clone tables and base tables of clone tables

In addition, NGT Unload does not unload temporal columns or write them to SYSPUNCH.

Sizing the work data set

During processing, the size required for the work data set (WK00001) is double the size of the SYSREC file. NGT Unload uses this disk space for internal sort and data staging or de-staging for multiple server processes.

For more information about the allocation of the WK00001 work data set, see the section in the *BMC Next Generation Technology Automation Reference Manual* that deals with the XSUTDBMG automation control point.

Selecting data to unload

This topic provides information that you need to know about selecting data to unload in NGT Unload.

Selecting columns

Use standard SQL SELECT statements to select columns in NGT Unload.
The following is a simple example of an NGT Unload SELECT clause:

```sql
UNLOAD FORMAT(UNLOAD)
SELECT COL01, COL02, 'CONSTANT', INTEGER(123)
WHERE COL02 = '20'
ORDER BY COL01
FROM TABLE NGT.UNLOAD_TEST
```

In the list of selected expressions, you can specify column names, functions, expressions, and constants, just as you would with SQL. You can also use the SQL CASE construct to apply logic to data and substitute values.

The following is an example using two CASE constructs:

```sql
UNLOAD FORMAT(DSNTIAUL)
SELECT COL01,
CASE COL02
  WHEN 'TX' THEN 'TEXAS'
  WHEN 'CA' THEN 'CALIFORNIA'
  WHEN 'MO' THEN 'MISSOURI'
  WHEN 'AK' THEN 'ALASKA'
  ELSE CASE
    WHEN COL02 BETWEEN '01' AND '50' THEN 'USA'
    WHEN COL02 = '51' THEN 'ITALY'
    WHEN COL02 = '52' THEN 'CANADA'
    ELSE NULL
  END
END,
'BMC',
INTEGER(123)
FROM TABLE NGT.UNLOAD_TEST
```

You can also use concatenation, substring operations, functions, and special registers. For more information, see the NGT SQL language chapter of the *BMC Next Generation Technology General User Guide*.

---

**Note**

NGT Unload supports SUBSELECT clauses used within the CASE construct.

---

**Selecting rows**

By default, NGT Unload unloads all table rows.

You can filter the rows to be unloaded using one of the following methods:

- Adding a WHERE clause
- Using the MAXROWS option
- *(for image copies)* Selecting with an alternate OBID

You can specify a WHERE clause and the MAXROWS option simultaneously. If you specify both, NGT Unload applies the WHERE clause first, then MAXROWS.
You can specify SQL WHERE clauses that contain subselects in NGT Unload. For more information about NGT Unload WHERE clauses, see the *BMC Next Generation Technology General User Guide*.

### Partition filtering

Use an SQL WHERE clause to select which rows to unload from a table.

For more information about WHERE clause specifications, see the *BMC Next Generation Technology General User Guide*.

NGT Unload automatically skips partitions that do not satisfy the provided WHERE clause criteria. This reduces the number of partitions that NGT Unload must read.

To be eligible for partition filtering, an unload must satisfy the following criteria:

- For both the UNLOAD and UNLD command:
  - The unloaded table is a partitioned table space.
  - The unload command has a WHERE clause that has at least one predicate that references the first key column of a limit key.

- For the UNLD command, the following options are not specified:
  - PART
  - COPYDSN
  - COPYDDN
  - NOFILTER

The NGT Unload output shows the total number of partitions and the partition numbers that satisfy the search criteria.

NGT Unload uses WHERE predicates to determine whether to skip a partition. NGT Unload supports the following three types of predicates for this purpose:

- **Basic predicate**

  For NGT Unload to analyze a basic predicate, it must be in one of the following formats:
  
  - `columnName relationalOperator constant`
  - `constant relationalOperator columnName`

  The variables are defined as follows:

  - `columnName` is a key column of the partitioning index.
— relationalOperator is a valid operator as described in the NGT SQL language chapter of the *BMC Next Generation Technology General User Guide*. NGT Unload analyzes all operators except ^= or <>. 

— *constant* is a constant or a special register that is free of any expressions.

### IN predicate

For NGT Unload to analyze the IN predicate, it must be in the following format:

```sql
columnName IN (constant, constant)
```

### BETWEEN predicate

For NGT Unload to analyze the BETWEEN predicate, it must be in one of the following formats:

- `columnName BETWEEN constant AND constant`
- `columnName NOT BETWEEN constant AND constant`

NGT Unload also analyzes the use of search condition operators (such as AND and OR) to reduce or expand the partition limit range. Using the AND operator in predicates reduces the number of partitions. Using the OR operator in predicates increases the selected partition range.

If a predicate does not conform to the preceding rules, NGT Unload selects all parts.

NGT Unload logically uses the AND operator with first key column conditions.

The partition selection analysis is exact when the partitioning index has a single-column key. For multiple-column key indexes, the analysis might result in an additional partition per range.

### Examples: single column key

In the following examples, the table (T1) is defined in a large partitioned table space with nine partitions.

T1 has two columns, both of TYPE data types C1 and C2. A partitioning index (CX1) indexes C1 in ascending order with the following limitkeys:

- Part 1 value(1000)
- Part 2 value(2000)
- Part 3 value(3000)
- Part 4 value(4000)
- Part 5 value(5000)
- Part 6 value(6000)
- Part 7 value(7000)
- Part 8 value(8000)
- Part 9 value(9000)

**Example 1**

```sql
UNLOAD FROM T1 WHERE (C1 > 1000 AND C1 <= 5000 AND C2+C1 = 6000)
```

The following messages appear:

- 4 PARTS SELECTED.
- PARTS 2-5

This means that NGT Unload reads partitions 2 through 5 only, and then evaluates the search criteria to select the applicable rows.

**Example 2**

```sql
UNLOAD FROM T1 WHERE (C1 > 1000 AND C1 < 5000 OR C2+C1=6000)
```

The following message appears:

- 9 PARTS SELECTED.

This means that NGT Unload selects all partitions because the predicate `C1+C2=6000` does not conform to the basic predicate rules.

**Example 3**

```sql
UNLOAD FROM T1 WHERE (C1 > 9000)
```

The following message appears:

- ALL PARTS EXCLUDED.

This means that NGT Unload does not select any partitions because the table space is LARGE and the last partition does not address keys less than 9000. If you attempt to insert into this table a key value greater than 9000, the SQL results in a DB2 error.

**Example 4**

```sql
UNLOAD FROM T1 WHERE (C1 > 1000 AND C1 < 5000 OR C1 >= 8000 AND C1 < 9000)
```

The following messages appear:

- 6 PARTS SELECTED.
- PARTS 2-5, 8-9

This means that NGT Unload reads partitions 2 through 5 and 8 through 9 only, and then evaluates the search criteria after reading the row data to ascertain the applicable rows.
Example 5

UNLOAD FROM T1 WHERE (C1 BETWEEN -100 AND 5000)

The following messages appear:
5 PARTS SELECTED.
PARTS 1-5

This means that NGT Unload reads the first five partitions.

Example 6

UNLOAD FROM T1 WHERE (C1 NOT BETWEEN 3500 AND 7500)

The following messages appear:
6 PARTS SELECTED.
PARTS 1-4, 8-9

This means that NGT Unload reads partitions 1, 2, 3, 4, 8, and 9.

Example 7

UNLOAD FROM T1 WHERE (C1 IN 1,5000,8000,10000))

The following messages appear:
3 PARTS SELECTED.
PARTS 1,5,8

This means that NGT Unload reads 1, 5, and 8.

Examples: multi-column key

In the following examples, the clustering index indexes columns C1 and C2 with the following limitkey values:
- Part 1 value(1000)
- Part 2 value(2000)
- Part 3 value(3000)
- Part 4 value(4000)
- Part 5 value(5000)
- Part 6 value(6000)
- Part 7 value(7000)
- Part 8 value(8000)
- Part 9 value(9000)
Example 1

UNLOAD FROM T1 WHERE (C1 > 1000 AND C1 <= 5000 AND C2+C1=6000)

The following messages appear:

5 PARTS SELECTED.
PARTS 2-6

This means that NGT Unload selects partition 6 (unlike in example 1 in Examples: single column key on page 21).

Example 2

UNLOAD FROM T1 WHERE (C1 BETWEEN -100 AND 5000)

The following messages appear:

6 PARTS SELECTED.
PARTS 1-6

This means that NGT Unload reads the first six partitions.

Example 3

UNLOAD FROM T1 WHERE (C1 IN (1,5000,8000,10000))

The following messages appear:

5 PARTS SELECTED.
PARTS 1,5-6,8-9

This means that NGT Unload reads partitions 1, 5, 6, 8, and 9.

Unloading from other sources

The procedure for unloading from a copy of a table space (an image copy, incremental copy, or a DSN1COPY) is similar to the procedure for unloading from a table space.

Specifying which copy to unload—using the UNLOAD command

To unload from a copy, use the INFILE option. For more information, see “INFILE” on page 68.
Specifying which copy to unload—using the UNLD command

To invoke this feature, add one of the following options to your unload statement:

**COPYDDN**

If you use COPYDDN, you must change the JCL to add the DD name supplied with the COPYDDN keyword. Either ensure that the DD in the JCL points to the input data set, or specify the OUTPUT command to provide the data set name.

For more information about the OUTPUT command, see the *BMC Next Generation Technology General User Guide*.

**COPYDSN**

If you do not want to change your JCL, use the COPYDSN keyword and supply the full data set name. Quotation marks around the data set name are optional.

**SYSCOPY**

Use the SYSCOPY keyword to unload from the last posted copy in the DB2 catalog, or to specify that you want the last FULL or INCREMENTAL copy.

Considerations for unloading from a copy

The attributes of the input data set that describe an image copy or DSN1COPY must be **RECFM=F**, or **RECFM=FB** and **LRECL=4096**. The input data set is assumed to contain valid DB2 page layouts.

When unloading from an image copy, NGT Unload counts, but does not use, control pages (that is, header pages and space map pages). Therefore, provided that the input data set contains valid DB2 pages, they can be in any order.

When unloading from an image copy, you can unload either from a single partition or all partitions. You cannot specify multiple partitions from which to unload.

Unloading from dropped tables

NGT Unload enables you to override the OBID value defined by DB2 when qualifying rows for unloading.
By default, NGT Unload uses the value that DB2 assigns. This makes it easier to recover data if a table is dropped and you need to unload from a copy of the table space.

Using the NGT Unload alternate OBID feature, you can override the OBID specification that NGT Unload retrieves from the DB2 catalog. This enables you, for example, to unload data from image copies or DSN1COPYS after a table has been dropped.

If you use an OBID value of 0, NGT Unload ignores all OBIDs in the input records. Use the value of 0 for single-table table spaces only. Using a value of 0 with a multi-table table spaces might produce unpredictable results or result in an abend.

**To use the alternate OBID feature**

1. Code the OBID keyword in the unload statement.

   Note the following considerations:

   - If you are using the UNLOAD command, you specify OBID on the FROM TABLE clause.
   - For the UNLD command, you must supply the OBID value as an integer. For the UNLOAD command, you can supply the value either as an integer or in hexadecimal format.
   - NGT Unload does not perform range validation of the supplied OBID.

2. If you are using the UNLOAD command, specify the INFILE option.

3. Ensure that the table name in the FROM TABLE clause in the unload statement exists. It need not contain rows, because NGT Unload needs only the column specification for the table.

4. Ensure that, for the table in the FROM TABLE clause, the columns that NGT Unload unloads are compatible with the table in the input copy data set.

5. If the table in the copy has an EDITPROC, ensure that the DB2 defined table is defined with that same EDITPROC.

**Saving space when using the DSNTIAUL format**

NGT Unload enables you to save output data set space when using the DSNTIAUL format.
To save space in this case, use the NOPAD option, or specify `DCB=RECFM=VB` in your JCL for the output data set. NGT Unload does not pad varying length fields.

The NOPAD option is valid only with the UNLD command.

**Page encoding (UNLD command)**

You can encode unloaded data in your UNLD command using a variety of code pages.

Encoding data applies only to single-byte character strings (SBCS) in fields defined as CHAR or VARCHAR. External numeric, date, and time fields are also subject to the encoding process. Character fields defined as BIT data are excluded. Double-byte character strings (DBCS) are also excluded.

Use one of the following options to specify how NGT Unload encodes unloaded data code pages. The CCSID option takes precedence over the others.

- CCSID (see [CCSID on page 131](#))
- ASCII (see [ASCII on page 130](#))
- UNICODE (see [UNICODE on page 153](#))
- EBCDIC (see [EBCDIC on page 134](#))

**Format delimiters**

Format delimiters are not subject to encoding. NGT Unload uses them as is, without any changes. You can override defaults with the appropriate encoded delimiters.

Format delimiters are as follows:

- NULLCHAR
- FIELDSEP
- CHARFLDDLM
- NULLSTRING
- COLDEL
- CHARDEL
- DECPT

The following table summarizes the behaviors of various data types. For additional information, see the NGT SQL information in the *BMC Next Generation Technology General User Guide*. 
Table 1: Data type behaviors

<table>
<thead>
<tr>
<th>Data type</th>
<th>Encoded?</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE, TIME, and TIMESTAMP formats</td>
<td>Yes</td>
<td>NGT Unload encodes all DATEFMT, TIMEFMT, and TSFMT skeletons from a CCSID of 37 to the desired CCSID.</td>
</tr>
<tr>
<td>HEX constants</td>
<td>No</td>
<td>NGT Unload encodes HEX constants in a SELECT statement and treats them as BIT data. For example: SELECT x'c1c2' from B</td>
</tr>
<tr>
<td>Padding</td>
<td>Yes</td>
<td>NGT Unload encodes pad characters. In FORMAT DSNTIAUL or FORMAT EXTERNAL, NGT Unload pads VARCHAR fields as follows: EBCDIC blanks in EBCDIC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ASCII blanks in ASCII</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unicode blanks in Unicode</td>
</tr>
</tbody>
</table>

Managing the generated LOAD statement data set

NGT Unload enables you to manage the creation of the LOAD statement in all output formats except UNLOAD. How you manage this depends on the unload command that you use.

UNLOAD command

When you use the UNLOAD command, the default DD, SYSCNTL, receives the LOAD statement. You can change this DD name or DD name prefix by using the CNTLDDN option.

You must include a SYSCNTL DD statement in your JCL (or a DD statement that matches the CNTLDDN option). The following considerations apply to SYSCNTL data sets:

- BMC recommends that you do not use a partitioned data set (PDS).
- If you specify multiple data sets, you must use the SYSCNTL*n form of the DD name.
- If you specify data control block (DCB) attributes, you must specify one of the following sets of attributes:
— For fixed-length data sets, RECFM can be F, FB, or FBS. The logical record length (LRECL) must be a minimum of 80 bytes.

— For variable-length data sets, RECFM can be V or VB. LRECL must be a minimum of 84 bytes.

- If you do not specify DCB attributes, NGT Unload uses DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120).

**UNLD command**

When you use the UNLD command, the default DD, SYSPUNCH, receives the LOAD statement. You can change this DD name with the STMTDDN option.

**Example**

To change the DD name to MYDD, specify the following unload statement:

```plaintext
//SYSIN DD *
UNLD DATA STMTDDN(MYDD) FROM TABLE tablename
//MYDD DD SYSOUT=X
```

The SYSPUNCH data set can have a record format (RECFM) of F, FB, V, VB, or U. The logical record length (LRECL) must be at least 80 bytes if fixed or undefined, or 84 bytes if variable. The block size must fit the requirements of the RECFM/LRECL combination. If the block size specified in the JCL is invalid for the RECFM/LRECL combination, NGT Unload overrides the block size and sets a correct block size.

When multiple unload statements are in the same SYSIN, either specify individual SYSPUNCH DD statements and supply a different member name on each DD statement, or specify DISP=MOD or DISP=OLD.

You can allocate SYSPUNCH to DD DUMMY or SYSOUT, and write it to TAPE or DASD.

If the DD name specified is not allocated, NGT Unload does not issue an error message and does not produce the LOAD statement.

**Server restrictions**

NGT Unload can affect your use of servers as follows:

- If you unload from an image copy using the COPYDDN keyword, the image copy data set is allocated and read in the master job, and servers cannot read the image copy data set. This restriction does not apply if you use the COPYDSN keyword or the OUTPUT command.
NGT Unload uses servers when no output DD statements are defined in the master job (subject to the preceding restriction). In these cases, the XULDDYNM automation control point or OUTPUT command dynamically names and allocates the unload data set.

If the output DD statement exists in the master job and the unload statement contains ORDER BY, the server can perform the read phase (subject to the first restriction) and NGT Unload performs the output phase in the master job in which the output data set is allocated.

### Differences between UNLOAD PLUS and NGT Unload

Review this topic if you are an existing user of UNLOAD PLUS and need to understand how NGT Unload 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 “UNLOAD syntax that is no longer valid” on page 128.</td>
</tr>
<tr>
<td>zIIP offload processing</td>
<td>NGT Unload 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>
<tr>
<td>Data sets: dynamic allocation</td>
<td>With NGT Unload, you can dynamically allocate your output and SYSCNTL (or SYSPUNCH) data sets by either using the XULDDYNM automation control point or by using the OUTPUT command. The OUTPUT command is similar to, but not exactly the same as, the UNLOAD PLUS 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 37.</td>
</tr>
<tr>
<td>Data sets: output</td>
<td>NGT Unload does not enable you to unload to both a primary and backup output file.</td>
</tr>
<tr>
<td>Copy input</td>
<td>The following differences apply to copy input:</td>
</tr>
<tr>
<td></td>
<td>- You cannot use a DDLIN data set for copy file input.</td>
</tr>
<tr>
<td></td>
<td>- NGT Unload does not enable you to use the following types of copy input:</td>
</tr>
<tr>
<td></td>
<td>- Inline copies</td>
</tr>
<tr>
<td></td>
<td>- Copies before the current one (-n copies)</td>
</tr>
<tr>
<td></td>
<td>- Encrypted copies created by the NGT Copy product</td>
</tr>
<tr>
<td></td>
<td>- Instant Snapshot copies created by NGT Copy</td>
</tr>
<tr>
<td></td>
<td>- VSAM linear data sets</td>
</tr>
<tr>
<td></td>
<td>- VSAM FlashCopy image copies</td>
</tr>
<tr>
<td></td>
<td>- Online consistent copies created by the BMC Recovery Management for DB2 or BMC Recovery for DB2 solutions</td>
</tr>
<tr>
<td>Functionality</td>
<td>Differences</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Spanned record</td>
<td>When you specify SPANNED YES, the output is standard spanned record format. This was not the case for UNLOAD PLUS.</td>
</tr>
<tr>
<td>format</td>
<td></td>
</tr>
<tr>
<td>WHERE clause</td>
<td>NGT Unload uses a more robust SQL-like language for MODE DIRECT (equivalent to DIRECT YES).</td>
</tr>
<tr>
<td>FORMAT</td>
<td>NGT Unload does not provide a BMC-internal format for moving data more quickly between NGT Unload and NGT Load.</td>
</tr>
<tr>
<td></td>
<td>NGT Unload does not support unloading to XML format.</td>
</tr>
<tr>
<td></td>
<td>To unload to delimited output, you must use the UNLD command with the FORMAT DELIMITED option. The UNLOAD command does not currently have a FORMAT CSV option.</td>
</tr>
<tr>
<td>Installation options</td>
<td>Instead of using the DOPTs installation options, NGT Unload uses configuration parameters to provide default processing values. You can also specify these parameters in your SYSIN.</td>
</tr>
<tr>
<td>Load control</td>
<td>NGT Unload does not produce load control statements specifically for use with the following products:</td>
</tr>
<tr>
<td>statements</td>
<td>■ Information Builder's FOCUS</td>
</tr>
<tr>
<td></td>
<td>■ SAS</td>
</tr>
<tr>
<td></td>
<td>■ CA Technologies' CA Easytrieve</td>
</tr>
<tr>
<td></td>
<td>■ Teradata</td>
</tr>
<tr>
<td></td>
<td>■ IBM's SQL/DS</td>
</tr>
<tr>
<td></td>
<td>■ Select Business Solution's NOMAD</td>
</tr>
</tbody>
</table>
Differences between UNLOAD PLUS and NGT Unload
Input for NGT Unload

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

DD statements

This topic describes NGT Unload DD statements.

SYSIN

This is the only DD statement required for use with NGT Unload. Specify utility statements as follows:

```
//SYSIN DD *
  (statement)
  (statement)
```

ULDPARMS

This DD statement is required only to override the NGT Unload parameters specified during installation. You can specify ULDPARMS in a data set or in the JCL. For example:

```
//ULDPARMS DD *
```

or

```
//ULDPARMS DD DISP=SHR,DSN=NGT.UNLOAD.PARMS
```
**UTLPARMS**

This DD statement is required only to override NGT utilities global parameters specified during installation. You can specify UTLPARMS in a data set or in the JCL. For example:

```
//UTLPARMS DD *
```

or

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

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

**SYSREC**

This DD statement names the output file. You can override this DD name using the UNLOADDDN option (on the UNLOAD command) or OUTDDN option (on the UNLD command). The default value is SYSREC, for the UNLOAD command, or SYSREC00, for the UNLD command. For example:

```
//SYSREC DD DISP=SHR,DSN=NEW.UNLOAD.OUTPUT.FILE
```

Instead of providing a data set name with the DD statement, you can dynamically allocate the data set by using one of the following methods. The value of the +OVERRIDEOUTPUT parameter determines which method NGT Unload uses.

- Providing the data set name in the XULDDYNM allocation control point
- Providing the data set name in an OUTPUT command in your unload SYSIN
  
  Place the OUTPUT command before the UNLOAD statement.

For more information about XULDDYNM, 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*.

**SYSCNTL or SYSPUNCH**

SYSCNTL (default for the UNLOAD command) or SYSPUNCH (default for the UNLD command) is the data set to which NGT Unload writes the output LOAD statement. You can write this statement to either a sequential data set or to a SYSOUT class. For example:

```
//SYSCNTL DD SYSOUT=* 
```
Instead of providing a data set name with the DD statement, you can dynamically allocate the data set by using one of the following methods. The value of the +OVERRIDEOUTPUT parameter determines which method NGT Unload uses.

- Providing the data set name in the XULDDYNM allocation control point
- Providing the data set name in an OUTPUT command in your unload SYSIN
  Place the OUTPUT command before the UNLOAD statement.

For more information about XULDDYNM, 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*.

**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 automation exits that you wish to use. This is an example of how this statement might appear in your JCL:

```plaintext
//NGTAUTO DD DISP=SHR,DSN=NGT.AUTOMATN.EXITS
```

For more information about NGT utilities global parameters, see the *BMC Next Generation Technology Automation Reference Manual*.

**NGT Unload syntax overview**

This version of NGT Unload enables you to use either the UNLOAD or UNLD command to run your unload jobs. UNLOAD is the preferred command and future enhancements will be made only to the UNLOAD command.

---

*Note*

The UNLD command will be deprecated in a future release.

---

Although many of the options are valid for both commands, if you use an option that is specific to one command with the other command, NGT Unload terminates with a syntax error. For example, you cannot use the OUTDDN option with the UNLOAD command, but you can use SPANNED YES with either command.
The UNLOAD command includes most of the options from the UNLOAD command in the BMC UNLOAD PLUS for DB2 product. For information about UNLOAD options that were valid in UNLOAD PLUS but are not valid for use with NGT Unload, see “UNLOAD syntax that is no longer valid” on page 128.

You can use the OUTPUT command with the UNLOAD command. For more information, see “OUTPUT command” on page 37.

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.
Specifying object names in your unload syntax

Note the following information about specifying object names in your unload syntax:

- You can use an alias or synonym wherever the syntax diagram shows a table name or view name.
- NGT Unload does not support Unicode table or view names.
- NGT Unload does not support delimited object names that do not have a character representation in EBCDIC.

OUTPUT command

You can use the OUTPUT command to dynamically allocate output and SYSCNTL data sets. The following considerations apply:

- You must specify the OUTPUT command before the UNLOAD statement.
- If you specify the OUTPUT command for an output data set and you also specify SPANNED YES, SPANNED YES takes precedence over the OUTPUT command.

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 output data sets, see “UNLOADDDN” on page 85.
- For information about using this command for SYSCNTL data sets, see “CNTLDDN” on page 55.

NGT Unload syntax diagrams

This topic contains the NGT Unload syntax diagrams. Options in lowercase italics (for example format option), indicate that there is a separate diagram for this option that follows the main syntax diagram for each command.
UNLOAD command diagrams

For option descriptions, see “NGT Unload option descriptions—UNLOAD syntax” on page 49.
UNLD command diagrams

For option descriptions, see “NGT Unload option descriptions—UNLD syntax” on page 129.
format option

mode option

select option

from table

order by option
Alphabetical listing of NGT Unload options

The following tables list the NGT Unload options and provide a reference to a description for each one.

**Table 2: NGT Unload options—UNLOAD syntax**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Reference</th>
<th>Valid with DIRECT NO (or DIRECT AUTO when it converts to DIRECT NO)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>“ASCII” on page 51</td>
<td>No</td>
</tr>
<tr>
<td>AUTOTAG</td>
<td>“AUTOTAG” on page 51</td>
<td>Yes</td>
</tr>
<tr>
<td>BLANKPAD</td>
<td>“BLANKPAD” on page 52</td>
<td>Yes</td>
</tr>
<tr>
<td>BYPART</td>
<td>“BYPART” on page 53</td>
<td>No</td>
</tr>
<tr>
<td>CCSID</td>
<td>“CCSID” on page 53</td>
<td>No</td>
</tr>
<tr>
<td>CNTLCARDS</td>
<td>“CNTLCARDS” on page 54</td>
<td>Yes</td>
</tr>
<tr>
<td>CNTLDDN</td>
<td>“CNTLDDN” on page 55</td>
<td>Yes</td>
</tr>
<tr>
<td>DATEFMT</td>
<td>“DATEFMT” on page 55</td>
<td>Yes</td>
</tr>
<tr>
<td>DELETEFILES</td>
<td>“DELETEFILES” on page 56</td>
<td>Yes</td>
</tr>
<tr>
<td>DIRECT</td>
<td>“DIRECT” on page 57</td>
<td>Yes</td>
</tr>
<tr>
<td>DISCARDS</td>
<td>“DISCARDS” on page 59</td>
<td>Yes</td>
</tr>
<tr>
<td>DRAIN_WAIT</td>
<td>“DRAIN_WAIT” on page 60</td>
<td>Ignored</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>“EBCDIC” on page 61</td>
<td>Yes</td>
</tr>
<tr>
<td>ESTROWS</td>
<td>“ESTROWS” on page 94</td>
<td>Yes</td>
</tr>
<tr>
<td>Keyword</td>
<td>Reference</td>
<td>Valid with DIRECT NO (or DIRECT AUTO when it converts to DIRECT NO)</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------------------------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td>EXCLUDE</td>
<td>“EXCLUDE PART” on page 62</td>
<td>Yes</td>
</tr>
<tr>
<td>FILL</td>
<td>“FILL” on page 62</td>
<td>Yes</td>
</tr>
<tr>
<td>FILTERPART</td>
<td>“FILTERPART” on page 63</td>
<td>Ignored</td>
</tr>
<tr>
<td>FIXEDVARCHAR</td>
<td>“FIXEDVARCHAR” on page 64</td>
<td>Yes</td>
</tr>
<tr>
<td>FORCE</td>
<td>“FORCE” on page 66</td>
<td>Ignored</td>
</tr>
<tr>
<td>FORMAT</td>
<td>“FORMAT” on page 66</td>
<td>Yes</td>
</tr>
<tr>
<td>FROM</td>
<td>“FROM” on page 89</td>
<td>Yes</td>
</tr>
<tr>
<td>INFILE</td>
<td>“INFILE” on page 68</td>
<td>No</td>
</tr>
<tr>
<td>INTO</td>
<td>“INTO” on page 88</td>
<td>Yes</td>
</tr>
<tr>
<td>LIMIT</td>
<td>“LIMIT” on page 72</td>
<td>Yes</td>
</tr>
<tr>
<td>LOGICAL PART</td>
<td>“PART or LOGICAL PART” on page 77</td>
<td>No</td>
</tr>
<tr>
<td>MINROWS</td>
<td>“MINROWS” on page 73</td>
<td>Yes</td>
</tr>
<tr>
<td>NOSUBS</td>
<td>“NOSUBS” on page 73</td>
<td>Ignored</td>
</tr>
<tr>
<td>NULLCHAR</td>
<td>“NULLCHAR” on page 73</td>
<td>Yes</td>
</tr>
<tr>
<td>NULLTYPE</td>
<td>“NULLTYPE” on page 74</td>
<td>Yes</td>
</tr>
<tr>
<td>OBID</td>
<td>“FROM” on page 89</td>
<td>Ignored</td>
</tr>
<tr>
<td>ON FAILURE</td>
<td>“ON FAILURE” on page 75</td>
<td>Yes</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>“OPTIONS” on page 94</td>
<td>No</td>
</tr>
<tr>
<td>ORDER</td>
<td>“ORDER” on page 76</td>
<td>Ignored</td>
</tr>
<tr>
<td>ORDER BY</td>
<td>“ORDER BY” on page 93</td>
<td>Yes</td>
</tr>
<tr>
<td>PART</td>
<td>“PART or LOGICAL PART” on page 77</td>
<td>No</td>
</tr>
<tr>
<td>RECORDID</td>
<td>“INTO” on page 88</td>
<td>Yes</td>
</tr>
<tr>
<td>RETRY</td>
<td>“RETRY” on page 78</td>
<td>Ignored</td>
</tr>
<tr>
<td>RETRY_DELAY</td>
<td>“RETRY_DELAY” on page 78</td>
<td>Ignored</td>
</tr>
<tr>
<td>SELECT</td>
<td>“SELECT” on page 86</td>
<td>Yes, with documented exceptions</td>
</tr>
<tr>
<td>SHRLEVEL</td>
<td>“SHRLEVEL” on page 79</td>
<td>Ignored</td>
</tr>
<tr>
<td>SPANNED</td>
<td>“SPANNED” on page 80</td>
<td>No</td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>“UNLOAD TABLESPACE” on page 50</td>
<td>No</td>
</tr>
<tr>
<td>Keyword</td>
<td>Reference</td>
<td>Valid with DIRECT NO (or DIRECT AUTO when it converts to DIRECT NO)</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>TIMEFMT</td>
<td>“TIMEFMT” on page 81</td>
<td>Yes</td>
</tr>
<tr>
<td>TSFMT</td>
<td>“TSFMT” on page 83</td>
<td>Yes</td>
</tr>
<tr>
<td>UNICODE</td>
<td>“UNICODE” on page 84</td>
<td>No</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>“UNLOAD” on page 50</td>
<td>Yes</td>
</tr>
<tr>
<td>UNLOADDN</td>
<td>“UNLOADDN” on page 85</td>
<td>Yes</td>
</tr>
<tr>
<td>ZONEDDECOVP</td>
<td>“ZONEDDECOVP” on page 85</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 3: NGT Unload options—UNLD syntax

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPEND</td>
<td>“APPEND” on page 130</td>
</tr>
<tr>
<td>ASCII</td>
<td>“ASCII” on page 130</td>
</tr>
<tr>
<td>AUTO</td>
<td>“MODE” on page 140</td>
</tr>
<tr>
<td>BLANKPAD</td>
<td>“BLANKPAD” on page 130</td>
</tr>
<tr>
<td>BYPART</td>
<td>“BYPART” on page 131</td>
</tr>
<tr>
<td>CCSID</td>
<td>“CCSID” on page 131</td>
</tr>
<tr>
<td>CLUSTER</td>
<td>“ORDER BY” on page 145</td>
</tr>
<tr>
<td>COPYDDN</td>
<td>“COPYDDN” on page 132</td>
</tr>
<tr>
<td>COPYDSN</td>
<td>“COPYDSN” on page 132</td>
</tr>
<tr>
<td>DATA</td>
<td>“DATA” on page 133</td>
</tr>
<tr>
<td>DATESFMT</td>
<td>“DATEFMT” on page 133</td>
</tr>
<tr>
<td>DB2SQL</td>
<td>“MODE” on page 140</td>
</tr>
<tr>
<td>DIGITS</td>
<td>“DIGITS” on page 134</td>
</tr>
<tr>
<td>DIRECT</td>
<td>“MODE” on page 140</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>“EBCDIC” on page 134</td>
</tr>
<tr>
<td>EXCLUDE</td>
<td>“EXCLUDE PARTS” on page 134</td>
</tr>
<tr>
<td>FORMAT</td>
<td>“FORMAT” on page 135</td>
</tr>
<tr>
<td>FROM[TABLE]</td>
<td>“FROM [TABLE]” on page 137</td>
</tr>
<tr>
<td>LOADSTMT</td>
<td>“LOADSTMT” on page 138</td>
</tr>
<tr>
<td>MAXERR</td>
<td>“MAXERR” on page 139</td>
</tr>
<tr>
<td>MAXLRECL</td>
<td>“MAXLRECL” on page 139</td>
</tr>
</tbody>
</table>
## NGT Unload option descriptions—UNLOAD syntax

The following topics describe the options that are available with the UNLOAD command.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXROWS</td>
<td>“MAXROWS” on page 139</td>
</tr>
<tr>
<td>MINROWS</td>
<td>“MINROWS” on page 73</td>
</tr>
<tr>
<td>MODE</td>
<td>“MODE” on page 140</td>
</tr>
<tr>
<td>MRF</td>
<td>“MODE” on page 140</td>
</tr>
<tr>
<td>NOFILTER</td>
<td>“NOFILTER” on page 145</td>
</tr>
<tr>
<td>NULLCHAR</td>
<td>“FORMAT” on page 135</td>
</tr>
<tr>
<td>OBID</td>
<td>“OBID” on page 145</td>
</tr>
<tr>
<td>ORDER BY</td>
<td>“ORDER BY” on page 145</td>
</tr>
<tr>
<td>OUTDDN</td>
<td>“OUTDDN” on page 146</td>
</tr>
<tr>
<td>PAD and NOPAD</td>
<td>“PAD or NOPAD” on page 147</td>
</tr>
<tr>
<td>PARTS and LPARTS</td>
<td>“PARTS or LPARTS” on page 147</td>
</tr>
<tr>
<td>ROWEXIT</td>
<td>“ROWEXIT” on page 148</td>
</tr>
<tr>
<td>SELECT</td>
<td>“SELECT” on page 148</td>
</tr>
<tr>
<td>SHRLEVEL CHANGE</td>
<td>“SHRLEVEL CHANGE” on page 149</td>
</tr>
<tr>
<td>SPANNED YES</td>
<td>“SPANNED YES” on page 149</td>
</tr>
<tr>
<td>STMFDN</td>
<td>“STMFDN” on page 150</td>
</tr>
<tr>
<td>SYSCOPY</td>
<td>“SYSCOPY” on page 151</td>
</tr>
<tr>
<td>TIMEFMT</td>
<td>“TIMEFMT” on page 151</td>
</tr>
<tr>
<td>TSFMT</td>
<td>“TSFMT” on page 152</td>
</tr>
<tr>
<td>UNICODE</td>
<td>“UNICODE” on page 153</td>
</tr>
<tr>
<td>WHERE clause</td>
<td>“WHERE” on page 154</td>
</tr>
<tr>
<td>ZEROPAD</td>
<td>“ZEROPAD” on page 154</td>
</tr>
</tbody>
</table>
UNLOAD

Use the UNLOAD command to unload data from one or more tables in a table space. In certain circumstances, you can use UNLOAD TABLESPACE instead to unload all tables in a table space.

UNLOAD TABLESPACE

databaseName.
tableSpaceName

UNLOAD TABLESPACE

databaseName.
tableSpaceName

This option is valid only when DIRECT YES is in effect.

NGT Unload generates a SELECT statement for each table in the table space. When you specify UNLOAD TABLESPACE, you obtain the same results as if you had specified SELECT * FROM for every table in the table space. Therefore, the rules for assigning the SYSREC and SYSCNTL output data sets are the same as if you had specified SELECT * FROM for every table in the table space.

This option can help you avoid maintenance of a large list of SELECT statements. When new tables are added to the table space or deleted from it, you need not change the syntax to unload the data.

Restrictions

The following restrictions apply to UNLOAD TABLESPACE:

- If you specify any of the following options with the UNLOAD TABLESPACE command, NGT Unload issues an error message and terminates:
  - DIRECT NO, or DIRECT AUTO that results in DIRECT NO processing
  - Any SELECT statement clause

- If you specify UNLOAD TABLESPACE when unloading table spaces that contain the following, NGT Unload terminates:
  - A LOB or XML column
--- A clone table

**Additional considerations**

The following considerations apply to the UNLOAD TABLESPACE command:

- The SELECT statements are generated in OBID order.
- If you do not specify the name of the database, NGT Unload uses the default database name DSNDB04.
- NGT Unload automatically uses AUTOTAG YES. For more information, see AUTOTAG on page 51.

**ASCII**

This option specifies that NGT Unload use the DB2 installation default (DSNHDEC) CCSIDs to encode the unloaded data in ASCII if you do not include a CCSID specification on your UNLOAD command.

Specifying the CCSID option overrides the ASCII option.

NGT Unload ignores this option if you also specify FORMAT UNLOAD.

**AUTOTAG**

Specify the AUTOTAG option to tell NGT Unload whether to add a four-byte character constant value at the beginning of each output record and to generate appropriate load control statements.

Specify AUTOTAG YES when the following conditions exist:

- You are using a single SYSCNTL data set.
You specify multiple SELECT statements on tables that are in the same table space.

When you specify UNLOAD TABLESPACE, NGT Unload automatically uses AUTOTAG YES, regardless of the value that you specify.

NO

Except when you specify UNLOAD TABLESPACE, AUTOTAG NO is the default. If you specify AUTOTAG NO, NGT Unload does not add a value to the beginning of each output record for each SELECT statement. When generating load control statements, NGT Unload generates syntax for multiple LOAD statements, generating one LOAD for each SELECT statement.

YES

If you specify AUTOTAG YES, NGT Unload adds a four-byte value to the beginning of each output record for each SELECT statement, and increments the value for each successive SELECT statement. For example, the value for the first SELECT statement is 0001; the value of a second SELECT statement would be 0002.

When generating load control statements, NGT Unload

- Generates syntax for one LOAD statement with multiple INTO clauses, generating one INTO clause for each SELECT statement
- Generates appropriate WHEN syntax

Considerations

The following considerations apply to AUTOTAG YES:

- You cannot specify AUTOTAG YES if you also specify FORMAT UNLOAD.
- RECORDID overrides AUTOTAG YES.

BLANKPAD

Use the BLANKPAD option to pad data in VARCHAR columns with blanks instead of low values.

BLANKPAD overrides the +BLANKPAD parameter value.
**BYPART**

BYPART tells NGT Unload to unload individually to separate files the partitions that you specify on the OPTIONS statement.

The BYPART option applies only when the target table to be unloaded is partitioned and dynamic allocation is in use. If not, NGT Unload ignores this option.

If you do not specify a range of partitions with BYPART, NGT Unload unloads the total number of partitions in the table space individually to separate files.

You cannot use the BYPART option in combination with the following options:

- DIRECT NO
- DIRECT AUTO
- PART (at the global level)

**CCSID**

This option specifies that NGT Unload use up to three CCSIDs to encode the unloaded data.

The three CCSIDs are for single-byte character set (SBCS), MIXED data, and double-byte character set data (DBCS) respectively. If you omit a CCSID value or specify 0, NGT Unload uses the corresponding DB2 system default CCSID for the encoding scheme that you specify or default to (EBCDIC, ASCII, or UNICODE).

BMC recommends that you use this option only to override your DB2 system default CCSID values.

*Restrictions*

NGT Unload supports translation from one CCSID to another with the following restrictions:

- Translations that require the use of a conversion procedure are not supported.
- NGT Unload terminates if you also specify DIRECT NO.
- NGT Unload ignores this option if you also specify FORMAT UNLOAD.
CNTLCARDS

The CNTLCARDS option specifies the type of control statements that NGT Unload writes to the SYSCNTL data set. You must specify a SYSCNTL data set in your JCL.

DB2LOAD

CNTLCARDS DB2LOAD is the default (when your JCL contains a SYSCNTL DD statement). This value tells NGT Unload to generate only DB2 LOAD utility control statements in the SYSCNTL data set.

Additional considerations

The following considerations apply to CNTLCARDS DB2LOAD:

- When you specify ENUMROWS as the first string in the CNTLCARDS, NGT Unload generates an ENUMROWS value ‘ENUMROWS (.nn)’ during the unload phase. This value indicates the number of rows that NGT Unload might unload.

- When you specify SPANNED YES, NGT Unload adds the FORMAT SPANNED YES option to the control cards that it generates.

- When you are unloading row ID columns and DIRECT NO is in effect, NGT Unload generates load control statements for those columns as if they were defined as GENERATED BY DEFAULT.

- When you are unloading a system-period temporal object, NGT Unload generates field specification control cards for temporal-specific fields.

- When you are unloading application-period temporal data, NGT Unload generates field specification control cards for business-begin and business-end fields.

'string'

Use this option to specify a string of additional LOAD syntax to insert into the IBM DB2 LOAD utility or BMC LOADPLUS product control statements. You can specify multiple strings of up to 72 bytes each. Enclose each string in single quotes. To embed a quoted string (a string enclosed by single quotation marks) within a string, add a single quotation mark before and after the quoted string (for example, `'CITY='''AUSTIN'''`).
Note
You can also insert additional load syntax to follow the INTO option for the IBM DB2 LOAD generated utility control statements. For more information, see “record options” on page 88.

CNTLDDN

The CNTLDDN option allows you to override the default ddname or the default ddname prefix of the data set that contains the control statements that NGT Unload generates. The default is SYSCNTL.

Instead of providing a data set name with the DD statement, you can dynamically allocate the data set by using one of the following methods. If you dynamically allocate the data set, the value of the OVERRIDEOUTPUT parameter determines the method that NGT Unload uses.

- Provide the data set name in the XULDDYNM automation control point.
- Provide the data set name in an OUTPUT command in your unload SYSIN.
  You must place the OUTPUT command before the UNLOAD statement in your SYSIN.

For more information about XULDDYNM, 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.

DATEFMT

The DATEFMT option allows you to override the default DB2 external data format.

This option also overrides your DB2 date exit routine and the ULDPARMS +DATEFMT parameter. You can use this option to port unloaded data to other relational databases.
Restriction

NGT Unload ignores this option if you also specify FORMAT UNLOAD.

('string')

This option allows you to specify a string of date format elements. Valid values are:

- MM for month
- DD for day
- YY or YYYY for year

You can specify these values in any order and delimit them with any character or number, or any series of characters or numbers, although NGT Unload does not require a delimiter. NGT Unload treats any other characters as constants in their specified position.

Example

If today is July 17, 2016, the character string YYYY.MM.DD results in the date format 2016.07.17.

Requirements

The following requirements apply to the string that you specify with the DATEFMT option:

- The string must begin with one of the valid date format elements.
- Only one occurrence of each value is valid.
- If you specify this option, you must specify at least one value or the unload job terminates.

DELETEFILES

The DELETEFILES option tells NGT Unload whether to delete the SYSREC and SORTWK files when the unload is unsuccessful.
**YES**

DELETEFILES YES tells NGT Unload to perform DELETEFILES processing, which differs depending on whether the unload job is running in a worklist environment:

- During DELETEFILES processing outside of a worklist, NGT Unload deletes all physical sequential data sets whose ddnames match the SYSREC and SORTWK ddnames or ddname prefixes.

- If you specify DELETEFILES YES in a worklist, NGT Unload deletes only the dynamically allocated SYSREC data sets.

**Restrictions**

NGT Unload does not delete the following files:

- Any files that are defined with an abnormal allocation disposition other than DELETE

- Any files that are defined with an expiration date or retention period that prevents the delete from being honored

NGT Unload does not delete files when the job terminates with an x22 abend.

**NO**

DELETEFILES NO tells NGT Unload not to delete any data sets after an unsuccessful unload.

**DIRECT**

The DIRECT option allows you to specify how NGT Unload processes table data.

- **YES**: Deletes all physical sequential data sets.
- **NO**: Does not delete any data sets.
- **AUTO**: Deletes data sets based on row set size.
- **ROWSETsz**: Specifies the row set size.
- **integer**: The integer value for the row set size.

**Requirements and restrictions**

The following requirements and restrictions apply to the DIRECT option:
■ In your SYSIN command stream, place any DIRECT option specification immediately after your UNLOAD or UNLOAD TABLESPACE option.

■ If you specify the DIRECT option more than once in your JCL, NGT Unload terminates.

■ You must specify DIRECT YES when you are performing data translation during the unload process.

■ You must specify DIRECT NO when you are unloading LOB or XML data to standard unload data sets (SYSREC).

Note
You can specify DIRECT YES to unload LOB or XML data to VBS unload data sets.

YES

If you specify DIRECT YES, NGT Unload uses its SELECT syntax to read table data directly from the table space data set. If the SELECT statement contains syntax that is not within the scope of the NGT Unload UNLOAD syntax, the utility terminates. This option provides high-performance unloads of DB2 table data, but might limit the functionality that the SELECT statement provides.

NGT Unload terminates when you specify DIRECT YES for any of the structures and data types that require DIRECT NO.

NO

If you specify DIRECT NO, NGT Unload uses DB2 dynamic SQL to process the SELECT statement and read the table data. The DIRECT NO option enables the full range of functionality that the DB2 SQL SELECT statement provides.

DIRECT NO is not a high-performance solution for unloading large volumes of data.

Considerations

Note the following restrictions and other considerations for the DIRECT NO option:

■ Dynamic SQL processing cannot occur on identity columns or ROWID data types that are defined as GENERATED ALWAYS. If generating load control statements and DIRECT NO is in effect, NGT Unload generates the control statements as if the column is defined as GENERATED BY DEFAULT.
NGT Unload does not support all options when you specify DIRECT NO. For example, NGT Unload ignores ORDER YES.

For a full list of the options that are not available with DIRECT NO, see the list of options in “Alphabetical listing of NGT Unload options” on page 46.

The default table name supplied in the INTO statement of the generated control statements is derived from the first table name in the SELECT statement. To generate control statements with the correct table name, use INTO NAME ownerName.tableName.

**ROWSETSZ**

The ROWSETSZ option allows you to tell NGT Unload how many rows to include in a rowset for a single FETCH request. You can specify one of the values described in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 or 1</td>
<td>Tells NGT Unload to fetch a single row at a time</td>
</tr>
<tr>
<td>2 through 32767</td>
<td>Tells NGT Unload to fetch the specified number of rows (as a rowset)</td>
</tr>
</tbody>
</table>

BMC recommends a value of 100 for most jobs.

**Restriction**

NGT Unload ignores ROWSETSZ when you are unloading LOB or XML data.

**AUTO**

If you specify DIRECT AUTO, NGT Unload chooses between processing the job as if you specified DIRECT YES, or processing it as if you specified DIRECT NO. If a SELECT statement contains statements that are not within the scope of the NGT Unload SELECT-like syntax, NGT Unload uses DB2 dynamic SQL to read the data when processing the SELECT statement.

To ensure consistent behavior of your NGT Unload jobs, specify DIRECT YES or DIRECT NO. NGT Unload enhancements could change the behavior of DIRECT AUTO.

**DISCARDS**

Specify the DISCARDS option to define the limit on the number of records that NGT Unload discards.
The default is DISCARDS 0 (no limit).

Records that NGT Unload discards for any reason count toward the limit that you specify with the DISCARDS option. NGT Unload terminates if it reaches the discard limit.

NGT Unload currently discards records only when the formatted record exceeds the LRECL of the output file.

NGT Unload does not write the discarded records to any data set.

**DRAIN_WAIT**

The DRAIN_WAIT option specifies the drain timeout value to use.

This option is meaningful only when DIRECT YES is in effect.

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

1. Releases the drains that it has obtained so far
2. Waits the length of time that you specify in the RETRY_DELAY command option
3. Tries again to drain the objects for the number of times that you specify in the RETRY command option

**Restriction**

NGT Unload ignores this option when DIRECT NO is in effect.
NONE

If you specify NONE, the drain request that NGT Unload issues times out immediately if the drain cannot acquire the lock. NONE prevents any application transactions from being queued during the drain process. BMC recommends that you specify NONE in high-transaction environments.

UTIL

UTIL tells NGT Unload to use the standard DB2 utility timeout value defined in DSNZPARMS for your site (IRLMRWT multiplied by UTIMOUT). The wait time applies to each object involved in the unload.

SQL

If you specify SQL, NGT Unload uses the standard SQL timeout value (IRLMRWT) as the drain timeout value. The wait time applies to each object involved in the unload.

integer

Specify any integer value from 0 through 1800, as follows:

- 0 is equivalent to the value UTIL.
- 1 through 1800 specifies the number of seconds to wait to obtain the drain for each drain retry before timing out.

EBCDIC

This option specifies that NGT Unload use the DB2 installation default (DSNHDECP) CCSIDs (coded character set identifiers) to encode the unloaded data in EBCDIC if you do not include a CCSID specification on your UNLOAD command. This option is the default for the three encoding types.

Specifying the CCSID option overrides the EBCDIC option.

NGT Unload ignores this option if you also specify FORMAT UNLOAD.
EXCLUDE PART

Use this option with the UNLOAD command to tell NGT Unload to exclude specified partitions from unload processing.

You can specify a list of partitions, a range of partitions, or both. To specify a range of partitions, use the format x:y, where y is greater than x.

The list of partitions or ranges does not have to be in order. However, the list cannot contain duplicate partitions. For example, the following list is valid:

EXCLUDE PART (10,2:3,6)

But the following list is invalid:

EXCLUDE PART (10,2:3,2:6)

FILL

The FILL option tells NGT Unload whether to fill numeric external fields with zeros when converting numeric columns to their external representations.

This capability supports applications such as COBOL that require leading zeros to edit and check unloaded data properly. You can use this global option or the FILL option on your field specification. For more information, see the individual data type.

Restriction

This option does not apply to conversions that involve the DECFLOAT data type.

NO

(default) Specify NO if you want to remove all leading zeros when converting numeric data types to their external representations.

YES

When you specify FILL YES, NGT Unload externalizes all leading zeros when converting numeric data types to their external representations. The first character in
the external representation of the number is either a minus sign ('-') or a plus sign ('+').

**FILTERPART**

This option allows NGT Unload to filter out partitions that do not meet the WHERE clause criteria when you use the first column of the partitioning key in your WHERE clause.

Specifying PART within a SELECT OPTIONS statement overrides the FILTERPART command option for the corresponding SELECT statement.

**Restriction**

NGT Unload ignores this option when DIRECT NO is in effect.

**NO**

FILTERPART NO is the default. When you specify FILTERPART NO, NGT Unload reads all partitions of the partitioned table space.

**YES**

When you specify FILTERPART YES, NGT Unload filters out partitions that do not meet the selection criteria. NGT Unload uses only the SYSREC data sets that are related to the unloaded partitions.
Example

In the following example, COL1 is the column that causes the break of partitions. If you use COL1 in the WHERE clause, NGT Unload filters out partitions that do not meet the criteria of the WHERE clause. If you specify FILTERPART YES, NGT Unload reads only partitions 2 and 3.

Three-column partitioning key:

<table>
<thead>
<tr>
<th>Partition number</th>
<th>COL1</th>
<th>COL2</th>
<th>COL3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partition 1</td>
<td>199</td>
<td>xxx</td>
<td>005</td>
</tr>
<tr>
<td>Partition 2</td>
<td>299</td>
<td>xxx</td>
<td>010</td>
</tr>
<tr>
<td>Partition 3</td>
<td>399</td>
<td>xxx</td>
<td>020</td>
</tr>
<tr>
<td>Partition 4</td>
<td>499</td>
<td>xxx</td>
<td>030</td>
</tr>
</tbody>
</table>

SELECT statement:

```sql
SELECT * FROM your.tableName
WHERE COL1 > '200'
AND COL1 < '399'
```

Considerations

The following restrictions and other considerations apply to FILTERPART YES:

- NGT Unload ignores this option when unloading image copies or partition-by-growth table spaces.
- For the predicate on your WHERE clause, if you specify a value that equals the limit key, NGT Unload might read an additional partition.

FIXEDVARCHAR

The FIXEDVARCHAR option allows you to specify whether to globally unload VARCHAR, VARGRAPHIC, and VARBINARY column values as variable-length or fixed-length fields when you do not explicitly specify a field type. Explicitly specifying a field type on the INTO statement overrides the FIXEDVARCHAR specification.

In either case, the field type is still VARCHAR, VARGRAPHIC, or VARBINARY. If you want any other field type, specify CHARACTER( n) as the field data type.
The default for this option depends on the value of the FORMAT option and the +PAD parameter:

<table>
<thead>
<tr>
<th>FORMAT option</th>
<th>+PAD parameter</th>
<th>FIXEDVARCHAR default</th>
</tr>
</thead>
<tbody>
<tr>
<td>No FORMAT option specified</td>
<td>Not applicable</td>
<td>NO</td>
</tr>
<tr>
<td>FORMAT DSNTIAUL</td>
<td>Not applicable</td>
<td>YES</td>
</tr>
<tr>
<td>FORMAT UNLOAD</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>FORMAT UNLOAD</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>All other FORMAT options</td>
<td>Not applicable</td>
<td>NO</td>
</tr>
</tbody>
</table>

**NO**

FIXEDVARCHAR NO indicates that VARCHAR, VARGRAPHIC, and VARBINARY columns are unloaded as variable-length fields without any padding before the beginning of the next field.

NGT Unload ignores this option when you specify FORMAT DSNTIAUL.

**YES**

When you specify FIXEDVARCHAR YES, NGT Unload performs the following actions:

- Unloads VARCHAR, VARGRAPHIC, and VARBINARY columns as fixed-length VARCHAR, VARGRAPHIC, and VARBINARY fields (unless you explicitly specify a field type)
  
  A fixed-length VARCHAR, VARGRAPHIC, or VARBINARY field has a length equal to the maximum length of the column + 2. The actual length specification of the column remains unchanged.

- Sets any unused bytes in the field (after the value and before the beginning of the next field) to blanks (VARCHAR and VARGRAPHIC) or hexadecimal zeros (VARBINARY)

- Sets the record format for the unload data set to FB

  **Exception:** If you specify RECFM VB on your UNLOAD command (or it is in effect by default), NGT Unload uses variable block format for the data set records.

**Restrictions**

Note the following restrictions when specifying FIXEDVARCHAR YES:

- NGT Unload does not change the 2-byte length indicator when you specify FIXEDVARCHAR YES. If you plan to load the unloaded data, use the control
statements that NGT Unload generates to ensure proper table loading. Otherwise, NGT Unload might load incorrect or truncated data.

- If you are unloading LOB or XML data, NGT Unload ignores this option for VBS data sets when you specify SPANNED YES.

**FORCE**

The FORCE option specifies whether to cancel DB2 threads that might prevent a drain process from completing. Currently, for NGT Unload, only FORCE NONE is available.

```
FORCE   NONE
```

FORCE NONE tells NGT Unload not to cancel DB2 threads that might prevent the drain process from completing.

**FORMAT**

The FORMAT option allows you to specify the format of unloaded data. All SELECT statements of all rows use the specified format.

```
FORMAT
EXTERNAL
DSNTIAUL
UNLOAD
```

You can override the specified format for a field by specifying an explicit format for the field.

For more information about these options, see “Output file formats” on page 167.

**EXTERNAL**

EXTERNAL tells NGT Unload to unload data for all data types in external format. If you do not specify a data type, NGT Unload uses the default length. Use this option only when applications require data in this format, or when you are loading the data to a non-MVS target and you cannot use one of the other format options.
NGT Unload terminates if you specify this option when you are unloading LOB or XML data.

**DSNTIAUL**

DSNTIAUL allows NGT Unload to use its high-performance direct access to produce an output file in which the unloaded data matches the output of the IBM DSNTIAUL program.

Unlike the DSNTIAUL program, NGT Unload (using DIRECT YES) supports ASCII and Unicode output.

When you specify FORMAT DSNTIAUL, NGT Unload performs the following functions:

- Provides data for all data types in internal format except for the date, time, and timestamp data types, which it provides in external format
- Pads all variable-length fields (VARCHAR, VARGRAPHIC, and VARBINARY) with X'00' to their maximum length
- Creates fixed-length output records with a maximum length of 32760 bytes

**Note**

If NGT Unload processes a LONG VARCHAR or LONG VARGRAPHIC column type, the maximum formatted row length may exceed the allowable output record length of 32760. You can use the INTO option with a field specification to reduce the length of the LONG column and avoid this error. To reduce the number of records that NGT Unload discards when they exceed the shortened length, specify the TRIM or TRUNCATE function to remove trailing spaces, or truncate the field to fit.

- Treats a string constant as variable length
- Puts a one-byte null indicator field after all nullable fields
  
  The null indicator field contains either X'6F' if the field is null or X'00' if it is not null.

**Restrictions**

Note the following restrictions when you specify FORMAT DSNTIAUL:

- You cannot use this option when unloading LOB or XML data. NGT Unload terminates in this case.
- When you specify FORMAT DSNTIAUL, NGT Unload sets the following options to the values shown:
— FIXEDVARCHAR to YES
— NULLCHAR to ‘?’
— NULLTYPE to T1

**UNLOAD**

The following considerations apply to FORMAT UNLOAD:

- Do not use DIRECT NO or DIRECT AUTO with FORMAT UNLOAD.
- You cannot specify an INTO statement when you specify FORMAT UNLOAD.
- If you specify a SELECT statement that uses a subset of columns in the table, you cannot reload the results into any table. Use this type of SELECT statement with FORMAT UNLOAD only to produce an output file that you can then reference in a WHERE or WHEN clause of a discard specification in NGT Reorg, as shown in the following example:

  DISCARD FROM TABLE tableName WHERE colName IN
  (FILE(fileName).FORMAT(UNLOAD))

**INFILE**

You can specify the INFILE option to use a copy or a data set other than the DB2 table space data set as the source of the input data. When you are unloading from multiple table spaces, the INFILE option applies to all table spaces.

This option is valid only when DIRECT YES is in effect.

The following table lists the alternate sources of input data that you can use, and tells you which INFILE option to use for each input type:

<table>
<thead>
<tr>
<th>Input type</th>
<th>INFILE option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full or incremental image copy data sets</td>
<td>IMAGECOPY or ddname</td>
</tr>
<tr>
<td>Cabinet copies created by NGT Copy or the BMC Recovery Management for DB2 solution or BMC Recovery for DB2 solution</td>
<td>IMAGECOPY</td>
</tr>
</tbody>
</table>
### Restrictions

The following additional restrictions apply to the `INFILE` option:

- You cannot specify `FILTERPART YES` when unloading from image copies.
- You cannot specify multiple partitions when unloading from image copies.
- NGT Unload does not unload from LOB or XML copy data sets. However, you can unload from a copy of the base table if you do not select any LOB or XML columns.
- You cannot use a DDLIN data set for copy file input.
- You cannot use the following types of copy input:
  - Inline image copies
  - Encrypted copies created by NGT Copy
  - Instant Snapshot copies created by NGT Copy
  - VSAM linear data sets
  - VSAM FlashCopy image copies
  - Online consistent copies created by the BMC Recovery Management for DB2 or BMC Recovery for DB2 solutions
- You cannot specify `-integer` to unload an image copy from `SYSIBM.SYSCOPY` other than the most current full or incremental copy.

### Additional considerations

The following additional considerations apply to the `INFILE` option:

- Unpredictable results might occur if the data sets do not match the options that you specify:
  - If the input data set is an incremental image copy, specify the `INCREMENTAL` option.
  - If you are unloading all partitions of a partitioned table space, ensure that the data set includes a full image copy of all of the partitions.
  - If you are unloading a specific partition, ensure that the input provides the specified partition. Provide the full image copy data set by using one `ddname`, or provide an image copy for the indicated partition.
When you specify INFILE to unload any of the following types of copy data sets, all selected rows must be at the current version (as defined in SYSIBM.SYSTABLES):

- An incremental image copy
- Any image copy created with the SYSTEMPAGES NO copy option

**IMAGECOPY**

Specify INFILE IMAGECOPY to have NGT Unload use the specified full or incremental image copy data set that is in the SYSIBM.SYSCOPY table or the BMCXCOPY table; BMCXCOPY applies only if you are unloading a cabinet copy.

For information that applies to all INFILE options, see the "Restrictions" and "Additional considerations" sections that follow the INFILE diagram.

**Note**

Cabinet copies are registered in BMCXCOPY as COPY_TYPE C.

If the table space is partitioned, NGT Unload uses either individual partition image copies (DSNUM is not equal to 0) or, if there is not one copy for each partition that you unload, NGT Unload uses the first image copy that fulfills your specifications and that contains all partitions (DSNUM is equal to 0).

NGT Unload can unload multi-data-set, nonpartitioned table spaces only from a single image copy that contains all of the individual data sets. You cannot unload nonpartitioned table spaces from separate image copy data sets where DSNUM is not equal to 0.

**Note**

NGT Unload might not unload from copies created with the SYSTEMPAGES NO option if those copies contain compression dictionaries created during DB2 SQL INSERT processing.

**FULL**

FULL, which is the default, tells NGT Unload to unload from a full image copy. NGT Unload unloads from the most recent full image copy found in the SYSIBM.SYSCOPY table or the BMCXCOPY table; BMCXCOPY applies only if you are unloading a cabinet copy.

**INCREMENTAL**

This option tells NGT Unload to unload from an incremental image copy. When unloading from an incremental image copy, NGT Unload performs the following actions:
Accesses the previous full copy for data set information (but does not unload from that copy)

Reads all rows in the incremental image copy (not only the rows that changed) and unloads those that match the SELECT statement criteria

NGT Unload unloads the most recent incremental image copy found in the SYSIBM.SYSCOPY table or the BMCXCOPY table; BMCXCOPY applies only if you are unloading a cabinet copy.

NGT Unload cannot use the INFILE option to unload incremental image copies that do not contain dictionary pages when the image copy contains compressed data.

**ddname**

Use this option to specify a ddname or ddname prefix when unloading from a DSN1COPY sequential data set or a specific full or incremental image copy.

**Note**

Do not use this option to unload from a cabinet copy; use INFILE IMAGECOPY instead.

If you specify INFILE *ddname*, NGT Unload unloads all rows from the specified data set that match your SELECT statement criteria.

**Additional restrictions and considerations**

In addition to the general restrictions and considerations on the INFILE option, the following restrictions and considerations apply to INFILE *ddname*:

- The following DB2 catalog information must match the copy that you are unloading:
  - The database identifier (DBID) and page set identifier (PSID) of the database and table space that contain the specified tables
  - The OBIDs of the tables that you specify, unless you specify the OBID for the table by using the OBID option
  - The table definitions
    However, if you added more columns to a table since the copy was made and you are unloading those columns, the new columns contain their default values.
- Concatenating multiple image copy data sets, particularly when mixing copy data sets from tape with copy data sets from DASD, produces unpredictable results.
When unloading an image copy on a DB2 subsystem other than the one on which the copy was created, the following information applies:

— For a range-partitioned table space, NGT Unload unloads only the number of partitions that are defined on the target subsystem.

— For a partition-by-growth table space, NGT Unload unloads only the number of partitions that exist on the target subsystem.

FULL

Specify this option to tell NGT Unload that the specified file is a full image copy or a DSN1COPY data set.

INCREMENTAL

Specify this option to tell NGT Unload that the specified file is an incremental image copy. If INFILE ddname specifies an incremental image copy, you must use this option or NGT Unload assumes that FULL (the default) applies, and unpredictable results might occur. When using an incremental image copy, NGT Unload reads all rows in that image copy, not only the rows that changed.

Note

NGT Unload cannot use the INFILE option to unload incremental image copies that do not contain dictionary pages when the image copy contains compressed data.

LIMIT

The LIMIT option defines the limit on the number of rows that NGT Unload selects from each table or partition of the table space.

The default is LIMIT 0 (no limit). If NGT Unload reaches the limit, it ends normally.

When unloading from a partitioned table space using DIRECT NO, LIMIT functions as if the table space were nonpartitioned.
**MINROWS**

Use this option to specify a minimum number of rows to unload. This option tells NGT Unload to issue a warning (RC=4) if the number of unloaded rows is less than the specified number.

You can specify any number greater than or equal to 0.

If you specify 0, then NGT Unload uses the global +MINROWS value specified in the ULDPARMS DD statement.

**NOSUBS**

Specify NOSUBS to have NGT Unload refuse to accept substitution characters during translation between CCSIDs.

*This option is meaningful only when DIRECT YES is in effect.*

If you specify NOSUBS and NGT Unload encounters a record that requires substitution, NGT Unload discards the record rather than unloading it.

**Restrictions**

NGT Unload ignores the option if you also specify FORMAT UNLOAD or DIRECT NO.

**NULLCHAR**

This option allows you to change the null indicator character in the output record.

With the exception of the HIVAL option, the values that you specify for the NULLCHAR option signify only a single byte. If the null indicator field is two bytes
long, NGT Unload propagates the constant to the second byte. The following table describes the values that you can specify for this option:

**Table 6: Values for the NULLCHAR option**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>This value tells NGT Unload to fill the null indicator field with the question mark character, '?'.</td>
</tr>
<tr>
<td>HIVAL</td>
<td>This value tells NGT Unload to fill the null indicator field with high values.</td>
</tr>
<tr>
<td>'c'</td>
<td>This value tells NGT Unload to fill the null indicator field with the character c, where c is any constant that is valid for C-type assembler language.</td>
</tr>
<tr>
<td>X'xx'</td>
<td>This value tells NGT Unload to fill the null indicator field with xx, where xx is any valid hexadecimal assembler constant, except X'00'.</td>
</tr>
</tbody>
</table>

If you specify FORMAT DSNTIAUL, NGT Unload overrides this value to '?'.

**NULLTYPE**

This option allows you to specify the location and length of the null indicator field in the output record.

The following table describes the values that you can specify for this option:

**Table 7: Values for the NULLTYPE option**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td><em>(default)</em> Specifies that the null indicator is 1 byte long, following the column.</td>
</tr>
<tr>
<td>T2</td>
<td>Specifies that the null indicator is 2 bytes long, following the column.</td>
</tr>
<tr>
<td>L1</td>
<td>Specifies that the null indicator is 1 byte long, preceding the column.</td>
</tr>
<tr>
<td>L2</td>
<td>Specifies that the null indicator is 2 bytes long, preceding the column.</td>
</tr>
</tbody>
</table>

**Considerations**

The following considerations apply to the NULLTYPE option:
If you specify SPANNED YES, this value applies only to the base table space. The LOB data will have a leading null indicator of X'FF' regardless of the value that you specify for this option.

NGT Unload ignores this option if you also specify FORMAT UNLOAD.

**ON FAILURE**

ON FAILURE tells NGT Unload how you want to handle an abnormal termination (when your job ends with a return code that is greater than or equal to 8, or abends).

The default for this option is ON FAILURE ALL TERMINATE UTILITY. In this case, if NGT Unload encounters a failure in any phase, the job terminates with the highest assigned return code.

**TERMINATE UTILITY**

When TERMINATE UTILITY is in effect for an abend that occurs during any or all of the UTILINIT, UNLOAD, or UTILTERM phases, NGT Unload completes the following tasks:

- Deletes the row that contains the utility ID from the BMCUTIL and BMCSYNC tables
- Issues the normal return code
- Deletes the SYSREC and SORTWK files if you have specified DELETEFILES YES

**Note**

If you are running the unload job in a worklist environment, NGT Unload deletes only dynamically allocated SYSREC data sets.

**Multiple statements**

When you specify multiple ON FAILURE statements, NGT Unload functions as follows:
For competing ON FAILURE statements, NGT Unload honors the last statement.

When a phase-specific ON FAILURE statement follows an ON FAILURE ALL statement, NGT Unload honors the phase-specific statement for that phase and the ON FAILURE ALL statement for remaining phases.

**ORDER**

Specify the ORDER option to tell NGT Unload whether to sort the output records.

ORDER NO is the default. ORDER NO unloads the output records in the same order in which NGT Unload found the data rows in the table space.

**YES**

This option is meaningful only when DIRECT YES is in effect.

ORDER YES sorts the selected rows in clustering key order or, if there is no clustering key, in partitioning key order. If a selected table does not have a clustering or partitioning key, NGT Unload changes this option to ORDER NO. If you are unloading a multi-table table space, NGT Unload sorts first by table, then by data-sorting key.

When multiple tasks unload rows to a single output data set, all tasks execute concurrently to perform the sorting. However, NGT Unload writes the selected rows in task sequence so that each output process completes before the next starts.

**Considerations**

The following considerations apply to ORDER YES:

The encoding scheme of the table determines the collating sequence for character data. For example, if the encoding scheme of the table is EBCDIC, the collating sequence is EBCDIC.

The ORDER and ORDER BY options are mutually exclusive. Use the ORDER BY option to order by anything other than the data-sorting key.
If you also specify SPANNED YES, NGT Unload does not sort LOB data and the LOB data is unloaded after the sort is complete. (NGT Unload sorts inline LOB data with the data.)

NGT Unload ignores this option if a clustering index does not exist on the table space.

When DIRECT NO is in effect, NGT Unload ignores ORDER YES and does not sort on the data-sorting key.

**PART or LOGICAL PART**

The PART option specifies the partition numbers of a partitioned table space to unload.

This option is valid only when DIRECT YES is in effect.

You can specify one, some, or all of the partitions either individually or by specifying a range. If you do not specify PART, NGT Unload unloads all partitions of the table space.

Optionally, you can specify the LOGICAL keyword with PART to indicate that you are specifying the logical partitions to unload rather than the physical partitions.

**Specifying partition numbers**

Note the following information about how to specify partition numbers with the PART option:

- Individual partitions in a list can be in any order (but partitions within a range must be in ascending order).

- You can specify a mixture of individual partitions and ranges of partitions.

- If you specify a partition number more than once, NGT Unload ignores any occurrence after the first.

- You can specify integer values from 1 through 4096.
The following example illustrates a valid PART specification for partitions 1 through 5, 7, and 10:

```
PART 10,1:5,7
```

**Considerations**

The following considerations apply to the PART option:

- You cannot specify multiple partitions when unloading from an image copy.
- When you are unloading XML data, the PART option applies only to the base table space.
- Specifying PART within a SELECT OPTIONS statement overrides the PART command option for that SELECT statement.
- Unloading objects with a large number of partitions increases the potential for encountering such issues as performance problems and memory restrictions.

**RETRY**

The RETRY option specifies the maximum number of times that you want NGT Unload to attempt to obtain a drain before it terminates.

```
RETRY integer
```

The number of attempts can range from 0 through 255.

NGT Unload ignores this option when DIRECT NO is in effect.

This option overrides the second value of the UTLPARMS +QRETRY parameter. The default for the second value of the +QRETRY parameter is 20.

**RETRY_DELAY**

After a drain times out, the RETRY_DELAY option specifies the minimum number of seconds that you want NGT Unload to wait before it tries again to obtain the drain.
The number of seconds can range from 1 through 1800.

NGT Unload ignores this option when DIRECT NO is in effect.

**SHRLEVEL**

The SHRLEVEL option specifies the level of access that concurrently operating DB2 applications and utilities have to the target spaces during unload processing.

**Restriction**

NGT Unload ignores this option when DIRECT NO is in effect.

**REFERENCE**

SHRLEVEL REFERENCE allows objects to remain in read/write access during unload processing, maintaining consistency with a point-in-time image of the data.

NGT Unload performs a drain of the writer on the objects that you are unloading. You can use the ULDPARMS +QRETRY parameter or the UNLOAD options DRAIN_WAIT, RETRY, and RETRY_DELAY to control how NGT Unload handles drain wait times and drain retries.

If you are unloading from a full or incremental image copy (including a cabinet copy), NGT Unload unloads only from image copies that are marked as SHRLEVEL REFERENCE in SYSIBM.SYSCOPY or, for a cabinet copy, the BMCXCOPY table.
CHANGE

If you are unloading from a table space, specify SHRLEVEL CHANGE to allow read/write access to the table space during unload processing.

If you are unloading from a full or incremental image copy (including a cabinet copy), NGT Unload unloads from image copies marked as either SHRLEVEL REFERENCE or SHRLEVEL CHANGE in SYSIBM.SYSCOPY or, for a cabinet copy, the BMCXCOPY table.

CONSISTENT NO

This option tells NGT Unload not to attempt to maintain consistency with a point-in-time image of the data when using SHRLEVEL CHANGE. When you specify CONSISTENT NO, NGT Unload does not restrict access to the object. CONSISTENT NO is the default for SHRLEVEL CHANGE. Be aware that specifying CONSISTENT NO might result in the following consequences:

- NGT Unload might not process any updated pages in the buffer pool.
- If you run NGT Unload on a DB2 object immediately following the creation of the data set for that object, any data involved might exist only in DB2 buffer pools, causing NGT Unload to terminate.
- If NGT Unload encounters any pages that have uncommitted data, it unloads the uncommitted data.

CONSISTENT YES

Specify CONSISTENT YES to have NGT Unload maintain consistency with a point-in-time image of the data while enabling read/write access to the data.

This option is not valid when you specify INFILE.

SPANNED

The SPANNED option tells NGT Unload whether you want to unload your LOB data to a data set in spanned-record format.
This option is valid only when DIRECT YES is in effect.

**NO**

NO, the default, tells NGT Unload not to unload LOB data to a data set in spanned-record format.

**YES**

YES tells NGT Unload to unload LOB data to a data set in spanned-record format.

The following general considerations apply when you unload to spanned-record data sets with SPANNED YES:

- NGT Unload terminates if you use SYSOUT=* as your SYSREC DD specification.
- NGT Unload overrides any record format or record size that you specify in your JCL for output data sets.
- The LOB columns must be listed at the end of the field specification.

The following considerations apply to the options that you specify when you unload to spanned-record data sets with SPANNED YES:

- NGT Unload ignores any translation specification and unloads the data using the encoding scheme of the current table. The LOAD statement that NGT Unload generates includes the CCSID option with the current encoding scheme.
- NGT Unload terminates when you specify SPANNED YES and any of the following options are also in effect:
  - ORDER YES
  - ORDER BY
- If you specify NULLTYPE with an L value, NGT Unload changes it to the comparable T value.
- When you specify SPANNED YES, NGT Unload ignores FIXEDVARCHAR YES.

**TIMEFMT**

The TIMEFMT option allows you to override the default DB2 external time format.
This option also overrides your DB2 time exit routine and the ULDPARMS +TIMEFMT parameter. This option is useful for porting unloaded data to other relational databases.

Restriction

NGT Unload ignores this option if you also specify FORMAT UNLOAD.

('string')

This option allows you to specify a string of time format elements. Valid values are:

- HH for hour
- MM for minute
- SS for second
- XM for AM or PM

You can specify these values in any order and delimit them with any character or number, or any series of characters or numbers, although NGT Unload does not require a delimiter.

Example

For a time of 3:14 in the afternoon, specifying the format string 'HH-MM(XM)' results in 3-14(PM).

Requirements

The following requirements apply to the string that you specify with the TIMEFMT option:

- The string must begin with one of the valid time format elements.
- Only one occurrence of each value is valid.
- If you specify this option, you must specify at least one value or the unload job terminates.

Additional considerations
The following considerations apply to this option:

- If you specify XM, NGT Unload converts the HH element to a 12-hour format.
- NGT Unload treats any other characters as constants in their specified position.

**TSFMT**

The TSFMT option allows you to override the default DB2 external timestamp format.

**Note**

There is no corresponding +TSFMT parameter in the ULDPARMS.

This option is useful for porting unloaded data to other relational databases.

**Restriction**

NGT Unload ignores this option if you also specify FORMAT UNLOAD.

**('string')**

This option allows you to specify a string of timestamp format elements. Valid values are

- DD for day
- MM or MO for month
- YY or YYYY for year
- HH for hour
- MM or MI for minute
- SS for second
- NN, NNNN, or NNNNNN for microsecond
- XM for AM or PM
You can specify these values in any order and delimit them with any character or number, or any series of characters or numbers, although NGT Unload does not require a delimiter.

Requirements

The following requirements apply to the string that you specify with the TSFMT option:

- The string must begin with one of the valid timestamp format elements.
- Only one occurrence of each value is valid.
- If you specify this option, you must specify at least one value or the unload job terminates.

Additional considerations

The following considerations apply to this option:

- You can specify MM for both months and minutes within one TSFMT option. NGT Unload determines whether MM designates months or minutes based on the position of MM within the string. If NGT Unload cannot determine whether MM designates months or minutes within the format, NGT Unload terminates.
- If you specify XM, NGT Unload converts the HH element to a 12-hour format.
- NGT Unload treats any other characters as constants in their specified position.

UNICODE

This option specifies that NGT Unload use the DB2 installation default (DSNHDECP) CCSIDs to encode the unloaded data in Unicode if you do not include a CCSID specification on your UNLOAD command.

Specifying the CCSID option overrides the UNICODE option.

When DIRECT YES is in effect, NGT Unload supports only UTF-8 (CCSID 1208) and UTF-16 (CCSID 1200) encoding.
UNLOADDN

The UNLOADDN option provides NGT Unload with a default ddname, prefix, or output descriptor for the unload data sets.

This option provides the following information:

- When unload data sets are allocated in the JCL, an override for the default ddname or prefix of those data sets
- When allocating unload data sets dynamically, the output descriptor or prefix to match to the OUTPUT statement

Multiple data sets

When specifying multiple unload data sets, the ddname that you specify becomes the ddname prefix. You must append \( n \) to the ddnames in the DD statements in your JCL, where \( n \) is a numeric value. These ddnames must not exceed eight characters.

If you specify more than 99 data sets, use the UNLOADDN command option to override the default data set name of SYSREC, specifying a ddname prefix that results in eight characters or less after appending the data set number.

Dynamic allocation

When NGT Unload dynamically allocates unload data sets, the \( outputDescriptor \) variable represents an output descriptor name or prefix. This name or prefix enables NGT Unload to match the dynamic allocation option values that you specify on the OUTPUT command with the correct unload data sets.

ZONEDDECOVP

Traditionally, NGT Unload formats decimal-zoned values with the system default zone digit, which is C for positive values and D for negative values. You can specify the ZONEDDECOVP option to assign overpunch values to decimal-zoned numeric values.
The overpunched values can be positive or negative.

- The first operand specifies the zone overpunch value for positive numbers. Standard positive overpunch values are A, C, E, and F.

- The second operand specifies the zone overpunch value for negative numbers. Standard negative overpunch values are B and D.

This option can be useful when you plan to use the unloaded data with an application that requires a specific zone value that is not the traditional default.

To obtain the absolute value of a number or to ensure that the value is in data external format, you can specify ZONEDDECOVP (F,F).

**SELECT statement options**

The following topics describe the options that you specify with the SELECT statement.

The options on the SELECT statement must be in order shown in the syntax diagram.

**SELECT**

Use the SELECT option to tell NGT Unload which columns to unload.
Restrictions

The following general restrictions apply to the SELECT option:

- You can specify only one SELECT option on your UNLOAD command.
- You cannot specify a SELECT statement when you also specify UNLOAD TABLESPACE.
- You cannot specify CURRENT RID.

Use one of the following methods to specify the columns to unload:

* (asterisk)

Specifying * tells NGT Unload to unload all columns from the specified table except the following columns:

- Hidden columns
- ROWID columns

This value is the default.

columnName

Specifying one or more column names tells NGT Unload to unload the indicated columns from the specified table.

expression

Specifying an expression tells NGT Unload to unload a subset of the columns from the specified table, or to unload the columns in an order that differs from original order of the specified table.

You can use expressions to refer to an operation on one or more columns.

For more information about using expressions in NGT utility products, see the *BMC Next Generation Technology General User Guide*. 
INTO

The INTO option defines the output record.

You can omit the INTO option entirely for a simple unload, in which case NGT Unload uses the default field data type, format, and length for all selected items.

You cannot specify an INTO statement when you specify FORMAT UNLOAD.

record options

The following options allow you to specify output record identification information.

NAME

The NAME option allows you to override the default table name or file that NGT Unload generates. When using DIRECT YES, the default name is the name of the table being unloaded. When using DIRECT NO, the table name is derived from the FROM option. To generate control statements with a name different from the default name, specify NAME ownerName.tableName.

Restriction

NGT Unload does not support Unicode table names.

'string'

This option allows you to insert additional load syntax to follow the INTO option for the IBM DB2 LOAD generated utility control statements when you specify CNTLCARDS DB2LOAD. You can specify multiple strings of up to 72 bytes each. Enclose each string in single quotes. To embed a quoted string (a string enclosed by single quotation marks) within a string, add a single
quotation mark before and after the quoted string. The following examples show the use of this option:

```
INTO 'PART 2:4 REPLACE'
INTO 'REPLACE WHEN EMPLODATE > ''2005-01-01'''
```

When loading system-period temporal objects, you might need to add syntax to the load statements that NGT Unload generates. You can either add the syntax as a string on the NGT Unload INTO option, or you can modify the load control cards that NGT Unload generated before you run the load job. The following table describes which strings to add for which conditions:

<table>
<thead>
<tr>
<th>Condition</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>You want to load input file data into a system-period begin or end column</td>
<td>'PERIODOVERRIDE'</td>
</tr>
<tr>
<td>You want to load input file data into a transaction-start-ID column</td>
<td>'TRANSIDOVERRIDE'</td>
</tr>
</tbody>
</table>

**RECORDID 'string'**

You can use the RECORDID option to specify a value for a record identifier. The RECORDID is a character string constant that can be up to 72 bytes long, and is placed at the front of each output record for the SELECT statement.

**Considerations**

The following considerations apply to the RECORDID option:

- If NGT Unload generates control cards for DB2, the utility generates appropriate WHEN syntax for the DB2 LOAD control statements.
- If you also specify AUTOTAG YES, NGT Unload ignores the AUTOTAG option for that SELECT statement.

**field specification**

The field specification defines the fields of the output record. See “field specification” on page 96 for a complete description, including restrictions.

**FROM**

FROM is a required keyword that specifies the table or view from which NGT Unload unloads data.
You can specify only one table or view name. If you do not specify the creator name, NGT Unload uses the DB2 primary authorization ID of the user executing NGT Unload. The table name or view name can be an alias or a synonym.

**Restrictions**

Note the following restrictions when specifying FROM:

- NGT Unload does not support the following specifications:
  - Three-part object names
  - Correlation names
  - Unicode table names

- You cannot specify an auxiliary or XML table. (To unload LOB or XML data, specify the base table.)

- NGT Unload does not support unloading from clone tables.

- With DIRECT YES, NGT Unload supports unloading from views except for the following definitions:
  - Views that you created with the keyword DISTINCT in the subselect
  - Views that you created with a GROUP BY clause in the subselect
  - Views that you created with a HAVING clause in the subselect
  - Views that you created with a join in the view subselect
  - Views that you created with a union in the view fullselect
  - Nested views

  **Note**
  DIRECT NO supports the full range of DB2 SQL SELECT functionality and view definitions.

- Only the authorization ID that created an unqualified synonym can access it.

**OBID**

*This option is meaningful only when DIRECT YES is in effect.*

The OBID option specifies the OBID that NGT Unload can use to find the rows for the selected table when unloading using the INFILE *ddname* option. Normally, NGT
Unload uses the OBID found in the DB2 catalog for the selected table to find the rows. This option allows you to override the OBID that NGT Unload finds in the DB2 catalog. This option is useful when the OBID in the DB2 catalog for the selected table no longer matches the OBID that is associated with the table as it exists in the image copy. This option is valid only when you specify the INFILE ddname option.

You can specify the OBID option when selecting data from tables or views. When NGT Unload uses the specified OBID with a view, the specified OBID overrides the OBID that NGT Unload found in the DB2 catalog for the "viewed" table. You can specify the OBID as an integer or as a hexadecimal string, and you must use either parentheses or braces.

**WHERE condition**

WHERE specifies the condition that must be true before NGT Unload can select a specific row.

A condition is a combination of predicates that use AND, OR, or () operators. NOT is not valid for an NGT Unload WHERE clause.

The following table shows the result of the WHERE condition when you use the AND or OR operator:

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>p AND q</th>
<th>p OR q</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>
NGT Unload evaluates predicates from left to right, with the following operator precedence:

- ()
- AND
- OR

When NGT Unload determines the resulting value of the condition, the utility stops further evaluation of any remaining predicates. If the result is true, NGT Unload selects the row. If the result is false, NGT Unload does not select the row.

**Restriction**

When unloading LOB or XML data, you can specify the WHERE clause only for a NULL condition.

**predicate**

A predicate is a simple comparison of a single column value to a constant or list of constants. The column name must be a column of the named table, and the length of the column must be less than or equal to 255.

NGT Unload does not support the ESCAPE keyword of the LIKE operator.

For predicate syntax information, see the NGT SQL language chapter of the *BMC Next Generation Technology General User Guide*.

**constants**

The constant specifies a value to be compared to the column value. NGT Unload then evaluates the result based on the preceding relative operator. For the comparison, the associated constant must match the data type of the column (that is, numeric to numeric, string to string, and date/time to date/time).

**Additional considerations**

The NGT SQL language chapter of the *BMC Next Generation Technology General User Guide* describes the requirements for constants in predicates. In addition, consider the following information:

- For the comparison, if the string is shorter than the column length, NGT Unload pads the string with blanks. However, NGT Unload processes VARCHAR comparisons like DB2 processes them.

- Although not prevented or restricted by DB2, BMC recommends that you avoid nonstandard comparisons. See rules for constants in the IBM DB2 *SQL Reference*. 
NULL

You can specify the keyword NULL as a constant for columns that are set to null. You can specify NULL with any of the allowed operators. The null value is always higher than all other values. When a column is set to NULL, a null indicator field follows each nullable field in the output.

Consideration

If a column value is NULL, a predicate (other than IS/IS NOT or where the constant is equal to NULL) does not evaluate to "unknown" as it does in DB2. Normally this produces the same result as in DB2, because NGT Unload treats an "unknown" value for a predicate as "false." However, using NOT with a predicate that evaluates to "unknown" is "true," which is different in DB2.

CURRENT DATE - labeledDuration

This option describes the current date or, optionally, the current date minus a duration in days, months, or years. The rules for date/time arithmetic using labeled durations are identical to the DB2 SQL rules. NGT Unload takes the current date from the local time-of-day clock during the UTILINIT phase.

CURRENT TIMESTAMP - labeledDuration

This option describes the current timestamp or, optionally, the current timestamp minus a duration in days, months, or years. The rules for date/time arithmetic using labeled durations are identical to the DB2 SQL rules. NGT Unload takes the current timestamp from the local time-of-day clock during the UTILINIT phase.

You cannot specify CURRENT TIMESTAMP WITH TIME ZONE. However, if you are comparing the current timestamp to a TIMESTAMP WITH TIME ZONE column, NGT Unload uses the time zone from the DSNHDECPI MPlicit_TIMEZONE value.

ORDER BY

Use the ORDER BY option to tell NGT Unload how to order the unloaded rows according to the specified columns.
Alternatively, if you are ordering by the data-sorting key, you can use the ORDER YES option. You cannot specify both ORDER YES and ORDER BY.

**Considerations**

The following considerations apply to the ORDER BY option:

- NGT Unload terminates when you specify a LOB or XML column on the ORDER BY option. (You can specify SPANNED YES as long as the ORDER BY option does not reference a LOB column.)

- The encoding scheme of the table determines the collating sequence for character data. For example, if the table is in EBCDIC, the collating sequence is EBCDIC.

Use one of the following specifications to indicate the order for the unloaded rows:

- ASC tells NGT Unload to sort the data in ascending order.
- DESC tells NGT Unload to sort the data in descending order.

**ESTROWS**

One of the methods that NGT Unload can use to calculate the size of the data sets that are produced during dynamic allocation is an ESTROWS number that you specify on a SELECT statement.

```
ESTROWS integer
```

*This option is meaningful only when DIRECT YES is in effect.*

The integer is the number of rows that you expect that NGT Unload will unload using that SELECT statement.

**OPTIONS**

OPTIONS provides support for options that apply to a particular SELECT statement.

```
OPTIONS - ( LOGICAL PART partitionNumber : partitionNumber )
```

*This option is meaningful only when DIRECT YES is in effect.*
Options specified within the SELECT OPTIONS statement override their global command option equivalents.

**PART or LOGICAL PART**

Specify PART followed by integer values from 1 through 4096, or a range of values. The values identify the partitions of the partitioned table space that NGT Unload will unload for the corresponding SELECT statement.

Optionally, you can specify the LOGICAL keyword with PART to indicate that you are specifying the logical partitions to unload rather than the physical partitions. NGT Unload terminates if you specify both logical and physical partitions in the same job.

Specifying PART within a SELECT OPTIONS statement overrides the PART and FILTERPART command options for the corresponding SELECT statement.

Use this option with BYPART to unload partitions individually to separate output files.

**Specifying partition numbers**

Note the following information about how to specify partition numbers with the PART option:

- Individual partitions in a list can be in any order (but partitions within a range must be in ascending order).
- You can specify a mixture of individual partitions and ranges of partitions.
- If you specify a partition number more than once, NGT Unload ignores any occurrence after the first.

In the following example, specifying PART 1, 3, 4, 5 within the SELECT OPTIONS statement overrides the PART 2 global command option. Only partitions 1, 3, 4, and 5 will be unloaded.

```sql
UNLOAD PART 2
SELECT * FROM MY.TABLE
OPTIONS (PART 1, 3, 4, 5);
```

You can also specify a range of values as shown in the following example. Specifying PART 1:5 (as shown in the following example) within the SELECT OPTIONS statement overrides the PART 2 global command option and unloads partitions 1, 2, 3, 4, and 5.

```sql
UNLOAD PART 2
SELECT * FROM MY.TABLE
OPTIONS (PART 1:5);
```
OPTIONS (PART 1:5) ;

**Dynamic allocation**

Specifying PART also determines which data sets NGT Unload dynamically allocates for the corresponding SELECT statement. In the following example, one data set will be generated for the SELECT statement (USER.P001) and only the first partition in the partitioned table space will be unloaded.

```
UNLOAD
UNLOADDN(SYSREC) ACTIVE (YES)
OUTPUT SYSREC DSNNAME 'USER.P&PART'
SELECT * FROM MY.PART3TB
OPTIONS (PART 1)
;  
```

; (semicolon)

Specify the semicolon to end the SELECT statement.

The semicolon is meaningful only when DIRECT YES is in effect, but is not required.

**field specification**

The field specification defines a field in the output record.
The field specification identifies the following information:

- The field name
- (optional) The data type, data format, and data length of the field
The value to assign to the field under certain conditions

You do not have to specify a field for each SELECT list item, but, if you do not include a field specification for an item, you must specify a comma (,) as a place holder. In this case, the description of the field defaults to that of the SELECT list item. If the INTO list has too many fields, NGT Unload terminates. If the INTO list has too few, the fields default to that of the SELECT list item.

Restrictions

Note the following restrictions when including a field specification:

- NAME and RECORDID are reserved words that you cannot use to name fields. If you want to use SQL reserved words or NGT Unload command options as object names, you must delimit them with quotation marks to prevent syntax errors.

- If you are using FORMAT UNLOAD, you cannot use the field specification list.

**fieldName**

The field name identifies a field in the output record. The field name does not have to match the corresponding column name in the SELECT list if a column name is used. If the column is nullable and you do not specify the IF NULL option, a one-byte character field follows immediately after the specified field. This one-byte field is set to X'00' if the value is not null and is set to a question mark (?) if it is null.

**dataType**

The data type option specifies the data type and format of the output value and can also indicate the length of the output value. If you do not specify the data type of the field, NGT Unload uses the data type and data length of the column or constant from the SELECT list. NGT Unload performs no conversions in this case, except for denormalization and FIELDPROC decoding.

For details about valid data types and general rules for data types, see “Data type keywords” on page 99. For allowable conversions, see “Supported data type conversions” on page 113.

**IF**

The IF option determines the value of the field based on the specified condition.

NGT Unload terminates when you specify the IF option and you are unloading LOB or XML data.
NULL

The IF NULL option determines the value of the field if the column value is null. The column must be nullable; if you specify IF NULL on the field, NGT Unload does not generate a null indicator on the field. If you use the IF NULL option, you cannot use the NULL value or any related predicates.

predicate

The IF predicate option determines the value of the field if NGT Unload determines that the predicate is true. See the description of the WHERE predicate option for rules about specifying the predicate.

VALUE

This option specifies that you want to place a constant value in the field if the IF condition is true. The constant must be of the same type as the column.

expression

You can use any valid SQL expression. For more information, see the BMC Next Generation Technology General User Guide.

NULL

This option specifies that you want a NULL value for the field if the condition is true. You may use NULL only if the corresponding column is nullable and you did not include IF NULL on the field specification.

CURRENT DATE

This option specifies the current date. The field must be the DATE data type.

CURRENT TIME

This option specifies the current time. The field must be the TIME data type.

CURRENT TIMESTAMP

This option specifies the current timestamp. The field must be the TIMESTAMP or TIMESTAMP WITH TIME ZONE data type.

Data type keywords

This section describes the keywords that you can use to specify data types on your field specification.
CHAR

CHAR (or CHARACTER) defines a fixed-length character field.

If you use CHAR on a column that is not defined as CHAR or VARCHAR, it is (almost) identical to specifying type EXTERNAL.

To ensure proper date formats when converting from a DATE column, BMC recommends that you set the ULDPARMS +DATEFMT parameter to DECP. To ensure proper time formats when converting from a TIME column, BMC recommends that you set the ULDPARMS +TIMEFMT parameter to DECP.

MIXED

MIXED defines a mixed character field. NGT Unload converts the target field values to the MIXED CCSID encoding of the source column.

FILL

FILL tells NGT Unload how to handle leading zeros when converting data from a source with numeric data types.

YES

FILL YES externalizes all leading zeros when converting numeric data types to their external representations. The first character in the external representation of the number is either a minus sign (-) or a plus sign (+). This capability supports applications such as COBOL that require leading zeros in order to edit and check unloaded data properly.

NO

FILL NO removes all leading zeros when converting numeric data types to their external representations.

(length)

This option specifies the number of single-byte characters. You can specify a length up to 255 bytes.

The following considerations apply to length specifications:

- Specifying a length shorter than the default can produce unpredictable results. For information about default lengths, see “Character, graphic, and binary output” on page 115.
When converting from a timestamp column and the specified length is greater than the default, NGT Unload pads the result with blanks.

For conversions from a CHAR or VARCHAR column, the following guidelines apply:

— If the column length is greater than the specified field length, either the source values must contain blanks for the difference, or you must specify TRUNCATE.

— If the column length is less than the specified field length, NGT Unload adds blanks to the resulting value for the length difference.

For conversions from a GRAPHIC or VARGRAPHIC column, the following guidelines apply:

— If the column length is greater than two times the specified field length, either the source values must contain DBCS blanks for the difference, or you must specify TRUNCATE.

— If the column length is less than two times the specified field length, NGT Unload adds blanks to the resulting value for the length difference.

TRIM

TRIM tells NGT Unload to remove as many trailing blanks from the string as needed to make the string length match the length that you specify.

NGT Unload performs the TRIM function before attempting to assign the value to the field. If the string is still too long, a conversion error occurs. NGT Unload applies TRIM before TRUNCATE.

TRIM is useful in the following scenarios:

■ When you are converting from a CHAR column to a VARCHAR field
■ When you are converting from a CHAR column to a shorter CHAR field, and you want to detect (by way of conversion error messages) nonblank data beyond position n

NGT Unload ignores TRIM when either of the following conditions exists:

■ Your input column and output field are both CHAR.
■ You are converting from GRAPHIC to CHAR.

TRUNCATE

TRUNCATE tells NGT Unload to truncate a string if it is longer than the field (even after the TRIM option has truncated trailing blanks). NGT Unload performs TRUNCATE after TRIM.
**VARCHAR**

VARCHAR defines a character string field that varies in length.

The following considerations apply to VARCHAR fields:

- The format of the field overrides FIXEDVARCHAR YES.
- To maintain consistency with your subsystem when converting from a DATE column, BMC recommends that you configure the ULDPARMS +DATEFMT parameter to DECP. To maintain consistency with your subsystem when converting from a TIME column, BMC recommends that you configure the ULDPARMS +TIMEFMT parameter to DECP.

**MIXED**

MIXED defines a mixed character field. NGT Unload converts the target field values to the MIXED CCSID encoding of the source column.

**FILL**

FILL tells NGT Unload how to handle leading zeros when converting data from a source with numeric data types. This capability supports applications such as COBOL that require leading zeros in order to edit and check unloaded data properly.

**YES**

When you specify FILL YES, NGT Unload externalizes all leading zeros when converting numeric data types to their external representations. The first character in the external representation of the number is either a minus sign (-) or a plus sign (+). This capability supports applications such as COBOL that require leading zeros in order to edit and check unloaded data properly.

**NO**

When you specify FILL NO, NGT Unload removes all leading zeros when converting numeric data types to their external representations.

**(length)**

The length indicates the number of single-byte characters. A 2-byte length field that contains the length of the character string in bytes precedes the VARCHAR string. This length does not include the two length bytes.

You can specify a length up to 255 bytes.
The following considerations apply to length specifications:

- Specifying a length shorter than the default can produce unpredictable results. For information about default lengths, see “Character, graphic, and binary output” on page 115.

- When converting from a timestamp column and the specified length is greater than the default, NGT Unload pads the result with blanks.

- For conversions from a CHAR or VARCHAR column, the following guidelines apply:
  
  - If the column length is greater than the specified field length, either the source values must contain blanks for the difference, or you must specify TRUNCATE.
  
  - If the column length is less than the specified field length, NGT Unload adds blanks to the resulting value for the length difference.

- For conversions from a GRAPHIC or VARGRAPHIC column when the column length is greater than two times the specified field length, either the source values must contain DBCS blanks for the difference, or you must specify TRUNCATE.

**TRIM**

TRIM tells NGT Unload to remove as many trailing blanks from the string as needed to make the string length match the length that you specify. If the string is still too long, a conversion error occurs. NGT Unload applies TRIM before TRUNCATE.

TRIM is useful when you are converting from a VARCHAR column to a CHAR(\(n\)) or VARCHAR(\(n\)) field and you want to detect (by way of conversion error messages) nonblank data beyond position \(n\).

TRIM does not remove trailing zeros when you are converting from a TIMESTAMP EXTERNAL or TIMESTAMP WITH TIME ZONE EXTERNAL field.

**TRUNCATE**

TRUNCATE tells NGT Unload to truncate a string if it is longer than the field (even after the TRIM option has truncated trailing blanks). NGT Unload performs TRUNCATE after TRIM.

**GRAPHIC** (\(length\))

This keyword defines a graphic field.
GRAPHIC indicates an internal graphic value without the shift-in and shift-out characters. Length indicates the number of double-byte characters.

For conversions from a CHAR or VARCHAR column, the following guidelines apply:

- If the column length is greater than two times the specified field length, either the source values must contain DBCS blanks for the difference, or you must specify TRUNCATE.

- If the column length is less than two times the specified field length, NGT Unload adds blanks to the resulting value for the length difference.

- The conversion uses the encoding scheme of the column.

For conversions from a GRAPHIC or VARGRAPHIC column, the following guidelines apply:

- If the column length is greater than the specified field length, either the source values must contain DBCS blanks for the difference, or you must specify TRUNCATE.

- If the column length is less than the specified field length, NGT Unload adds DBCS blanks to the resulting value for the length difference.

**TRUNCATE**

TRUNCATE tells NGT Unload to truncate a string if it is longer than the field.

**VARGRAPHIC (length)**

This keyword defines a variable-length graphic string field.

A 2-byte length field that contains the length of the graphic string in double-byte characters precedes the string. The length does not include the two length bytes.

The format of the field overrides FIXEDVARCHAR YES.

For conversions from a CHAR or VARCHAR column, the following guidelines apply:

- If the column length is greater than two times the specified field length, either the source values must contain DBCS blanks for the difference, or you must specify TRUNCATE.

- The conversion uses the encoding scheme of the column.
For conversions from a GRAPHIC or VARGRAPHIC column, the following guidelines apply:

- If the column length is greater than the specified field length, either the source values must contain DBCS blanks for the difference, or you must specify TRUNCATE.

- If the column length is less than the specified field length, NGT Unload sets the length of the field to the length of the column.

**TRUNCATE**

TRUNCATE tells NGT Unload to truncate a string if it is longer than the field.

**SMALLINT**

SMALLINT defines a small integer numeric field.

When converting from a character column, the source value can contain leading or trailing blanks.

**SMALLINT**(length, scale)

SMALLINT indicates an internal 2-byte integer value. If you specify the length, you must specify 2. Specifying scale multiplies or divides the number by a power of 10.

**SMALLINT EXTERNAL**(length, scale)

EXTERNAL indicates a string that contains a character representation of an integer constant. This data type is identical to INTEGER EXTERNAL.

**FILL**

FILL tells NGT Unload how to handle leading zeros when converting data to SMALLINT EXTERNAL. Specify YES or NO:

- FILL YES externalizes all leading zeros when converting numeric data types to their external representations. The first character in the external representation of the number is either a minus sign (-) or a plus sign (+). This capability supports applications such as COBOL that require leading zeros in order to edit and check unloaded data properly.

- FILL NO removes all leading zeros when converting numeric data types to their external representations.
**INTEGER**

This keyword defines an integer numeric field. You can abbreviate INTEGER to INT.

When converting from a character column, the source value can contain leading or trailing blanks.

**INTEGER**(length, scale)

INTEGER indicates an internal 4-byte integer value. If you specify the length, you must specify 4. Specifying scale multiplies or divides the number by a power of 10.

**INTEGER EXTERNAL**(length, scale)

EXTERNAL indicates a string that contains a character representation of an integer constant.

**FILL**

FILL tells NGT Unload how to handle leading zeros when converting data to INTEGER EXTERNAL. Specify YES or NO:

- FILL YES externalizes all leading zeros when converting numeric data types to their external representations. The first character in the external representation of the number is either a minus sign (-) or a plus sign (+). This capability supports applications such as COBOL that require leading zeros in order to edit and check unloaded data properly.

- FILL NO removes all leading zeros when converting numeric data types to their external representations.

**BIGINT**

BIGINT defines a big integer numeric field.

When converting from a character column, the source value can contain leading or trailing blanks.

**BIGINT**(length, scale)

BIGINT indicates an internal 8-byte integer value. If you specify the length, you must specify 8. Specifying scale multiplies or divides the number by a power of 10.
**BIGINT EXTERNAL**(*length, scale*)

EXTERNAL indicates a string that contains a character representation of an integer constant. This data type is identical to INTEGER EXTERNAL.

**FILL**

FILL tells NGT Unload how to handle leading zeros when converting data to BIGINT EXTERNAL. Specify YES or NO:

- **FILL YES** externalizes all leading zeros when converting numeric data types to their external representations. The first character in the external representation of the number is either a minus sign (-) or a plus sign (+). This capability supports applications such as COBOL that require leading zeros in order to edit and check unloaded data properly.

- **FILL NO** removes all leading zeros when converting numeric data types to their external representations.

**DECIMAL**

This keyword defines a decimal numeric field. You can abbreviate DECIMAL to DEC.

The following considerations apply to decimal field data types:

- When converting from a numeric column, the default precision for these fields is based on the source data type:
  - SMALLINT—5
  - INTEGER—11
  - BIGINT—19
  - DECFLOAT—31
  - All other data types—15

- When converting from a character column:
  - The source value can contain leading or trailing blanks.
  - Precision must be less than scale and scale cannot be greater than 31.
  - If the string contains a decimal point, NGT Unload converts the string based on that decimal point, then scales to the specified scale.

**DECIMAL PACKED**(*precision, scale*)

DECIMAL PACKED is the default, which indicates that the output value is packed decimal with precision and scale. The precision represents the total number of digits; the scale represents the number of digits in the fractional part of the number.
DECIMAL(*precision, scale*)

This option is identical to DECIMAL PACKED.

DECIMAL ZONED(*precision, scale*)

This option specifies that the output value is decimal-zoned with precision and scale. The precision represents the total number of digits; the scale represents the number of digits in the fractional part of the number.

DECIMAL EXTERNAL(*length, scale*)

EXTERNAL indicates a string that contains a character representation of a decimal constant. A decimal point is always placed in the string based on the scale specification.

FILL

FILL tells NGT Unload how to handle leading zeros when converting data to DECIMAL EXTERNAL. Specify YES or NO:

- FILL YES externalizes all leading zeros when converting numeric data types to their external representations. The first character in the external representation of the number is either a minus sign (-) or a plus sign (+). This capability supports applications such as COBOL that require leading zeros in order to edit and check unloaded data properly.
- FILL NO removes all leading zeros when converting numeric data types to their external representations.

FLOAT

FLOAT defines a floating-point numeric field.

When converting from a character column, the source value can contain leading or trailing blanks.

FLOAT(*length, scale*)

If you specify the length, the value must be in one of the following ranges:

- For a 4-byte internal floating-point number, the value must be between 1 and 21.
- For an 8-byte internal floating-point number, the value must be between 22 and 53.
FLOAT EXTERNAL(*length, scale*)

EXTERNAL indicates a string that contains a character representation of a float constant.

DECFLOAT

This option defines a decimal floating-point numeric field.

When converting from a character column, the source value can contain leading or trailing blanks.

(*precision*)

The precision represents the total number of digits. You can specify 16 (an 8-byte number) or 34 (a 16-byte number). If you do not specify a precision, NGT Unload uses the following defaults:

- If the source column is defined as data type DECFLOAT, NGT Unload uses the precision attribute of the source column.
- If the source column is not defined as data type DECFLOAT, NGT Unload uses a precision of 34 bytes.

EXTERNAL

EXTERNAL indicates a string that contains a character representation of a decimal floating-point constant.

(*length*)

The length indicates the number of single-byte characters. Valid values are 1 through 42. If you do not specify a value, NGT Unload defaults to one of the following values:

- 23 if the source column is defined as data type DECFLOAT with a length of 8 bytes
- 42 in the following cases:
  - If the source column is defined as data type DECFLOAT with a length of 16 bytes
  - If the source column is not defined as data type DECFLOAT
When converting from DECFLOAT to DECFLOAT EXTERNAL, the length that you specify must be long enough to contain the significant digits of the input value without rounding. Otherwise, NGT Unload discards the row.

**BINARY**

This keyword defines a fixed-length binary field.

*(length)*

This option specifies the number of single-byte characters.

**TRIM**

TRIM tells NGT Unload to remove as many trailing hexadecimal zeros from the string as needed to make the string length match the length you specify.

NGT Unload performs the TRIM function before attempting to assign the value to the field. If the string is still too long, a conversion error occurs. NGT Unload applies TRIM before TRUNCATE.

**TRUNCATE**

TRUNCATE tells NGT Unload to truncate a string if it is longer than the field (even after the TRIM option has truncated trailing hexadecimal zeros). NGT Unload performs TRUNCATE after TRIM.

**VARBINARY**

This keyword defines a binary string field that varies in length.

You can also specify this field type as BINARY VARYING.

*(length)*

The length indicates the number of single-byte characters. A 2-byte length field that contains the length of the character string in bytes precedes the VARBINARY string. This length does not include the two length bytes.

**TRIM**

TRIM tells NGT Unload to remove as many trailing hexadecimal zeros from the string as needed to make the string length match the length you specify. If the string is still too long, a conversion error occurs. NGT Unload applies TRIM before TRUNCATE.
**TRUNCATE**

TRUNCATE tells NGT Unload to truncate a string if it is longer than the field (even after the TRIM option has truncated trailing hexadecimal zeros). NGT Unload performs TRUNCATE after TRIM.

**DATE EXTERNAL**(length)

This keyword defines a date field as a DB2 date string.

To maintain consistency with your subsystem, BMC recommends that you configure the ULDPARMS +DATEFMT parameter to DECP.

**TIME EXTERNAL**(length)

This keyword specifies a time field as a DB2 time string value.

To maintain consistency with your subsystem, BMC recommends that you configure the ULDPARMS +TIMEFMT parameter to DECP.

**TIMESTAMP**

This keyword specifies a timestamp field and must be specified as one of the following TIMESTAMP specifications:

**TIMESTAMP EXTERNAL**(length)

This option specifies a DB2 timestamp string value.

Precision on a TIMESTAMP EXTERNAL field is the specified length minus 20. The following additional information applies to the precision of the timestamp value:

- If the length is less than 21, NGT Unload sets the precision to 0 and the length to 19.
- If the length is greater than 32, NGT Unload sets the precision to 12 and pads the result with blanks.
- If the length of the TIMESTAMP column is greater than the calculated precision of the TIMESTAMP field, NGT Unload truncates the seconds fractions.
- If the length of the TIMESTAMP column is less than the calculated precision of the TIMESTAMP field, NGT Unload extends the seconds fractions with zeros.
**TIMESTAMP WITH TIME ZONE EXTERNAL**(length)

This option specifies a DB2 timestamp with a time zone string value. NGT Unload extracts the time zone from DSNHDECP, if specified, or from the IBM z/OS communication vector table (CVT).

Precision on a TIMESTAMP WITH TIME ZONE EXTERNAL field is the specified length minus 26. The following additional information applies to the precision of the timestamp value:

- If the length is less than 27, NGT Unload sets the precision to 0 and the length to 25.
- If the length is greater than 38, NGT Unload sets the precision to 12 and pads the result with blanks.
- If the length of the TIMESTAMP column is greater than the calculated precision of the TIMESTAMP field, NGT Unload truncates the seconds fractions.
- If the length of the TIMESTAMP column is less than the calculated precision of the TIMESTAMP field, NGT Unload extends the seconds fractions with zeros.

**BLOB**

BLOB defines a field that contains binary large object (BLOB) data.

A four-byte length field precedes the string. This length does not include the four length bytes.

**CLOB**

CLOB defines a field that contains character large object (CLOB) data.

A four-byte length field precedes the string. This length does not include the four length bytes.

**DBCLOB**

DBCLOB defines a field that contains double-byte character large object (DBCLOB) data.

A four-byte length field precedes the string. This length does not include the four length bytes.
Supported data type conversions

The following sections describe the allowable data conversions and default output lengths.

A blank cell in a table indicates that the data type conversion is not allowed.

Numeric output

The following table lists the conversions that NGT Unload supports to numeric output. The table that follows this one lists the default output lengths for these conversions.

Table 10: Allowable data type conversions for numeric output fields

<table>
<thead>
<tr>
<th>Output field</th>
<th>SMALLINT</th>
<th>INTEGER</th>
<th>BIGINT</th>
<th>SMALLINT, INTEGER, or BIGINT</th>
<th>DECIMAL</th>
<th>DECIMAL ZONED</th>
<th>FLOAT</th>
<th>DECFLOAT</th>
<th>DECIMAL, FLOAT, or DECFLOAT EXTERNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input column</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td>D</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>INTEGER</td>
<td>X</td>
<td>D</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>BIGINT</td>
<td>X</td>
<td>X</td>
<td>D</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>D</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>FLOAT</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>D</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DECFLOAT</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>D</td>
<td>X</td>
</tr>
<tr>
<td>CHAR</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The following considerations apply to the default output length information for these fields:

- With FORMAT EXTERNAL, the length defaults to the following value unless EXTERNAL is specified on the field:
  - SMALLINT defaults to 6.
  - INTEGER defaults to 11.
  - BIGINT defaults to 20.
  - DECIMAL defaults to 17.

- For DECFLOAT columns converting to DECFLOAT EXTERNAL output fields, the default length depends on the source column:
  - If the source column is defined with a length of 8 bytes, the default is 23.
  - If the source column is defined with a length of 16 bytes, the default is 42.

Table 11: Default lengths for numeric output fields

<table>
<thead>
<tr>
<th>Output field</th>
<th>SMALLINT</th>
<th>INTEGER</th>
<th>BIGINT</th>
<th>DECIMAL, INTEGER, or BIGINT EXTERNAL</th>
<th>DECIMAL</th>
<th>DECIMAL ZONED</th>
<th>FLOAT</th>
<th>DECIMAL or FLOAT EXTERNAL</th>
<th>DECFLOAT</th>
<th>DECFLOAT EXTERNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input column</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>6, 11, 20 respectively</td>
<td>C</td>
<td>C</td>
<td>8</td>
<td>C+2, 24 respectively</td>
<td>16</td>
<td>42</td>
</tr>
<tr>
<td>INTEGER</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>6, 11, 20 respectively</td>
<td>C</td>
<td>C</td>
<td>8</td>
<td>C+2, 24 respectively</td>
<td>16</td>
<td>42</td>
</tr>
<tr>
<td>BIGINT</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>6, 11, 20 respectively</td>
<td>C</td>
<td>C</td>
<td>8</td>
<td>C+2, 24 respectively</td>
<td>16</td>
<td>42</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>6, 11, 20 respectively</td>
<td>C</td>
<td>C</td>
<td>8</td>
<td>C+2, 24 respectively</td>
<td>16</td>
<td>42</td>
</tr>
<tr>
<td>FLOAT</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>6, 11, 20 respectively</td>
<td>C</td>
<td>C</td>
<td>8</td>
<td>C+2, 24 respectively</td>
<td>16</td>
<td>42</td>
</tr>
<tr>
<td>DECFLOAT</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>6, 11, 20 respectively</td>
<td>C</td>
<td>C</td>
<td>8</td>
<td>C+2, 24 respectively</td>
<td>16</td>
<td>42</td>
</tr>
<tr>
<td>Output field</td>
<td>SMALLINT</td>
<td>INTEGER</td>
<td>BIGINT</td>
<td>SMALLINT, INTEGER, or BIGINT EXTERNAL</td>
<td>DECIMAL</td>
<td>DECIMAL ZONED</td>
<td>FLOAT</td>
<td>DECIMAL or FLOAT EXTERNAL</td>
<td>DECFLOAT</td>
<td>DECFLOAT EXTERNAL</td>
</tr>
<tr>
<td>--------------</td>
<td>----------</td>
<td>---------</td>
<td>--------</td>
<td>----------------------------------------</td>
<td>---------</td>
<td>---------------</td>
<td>-------</td>
<td>-----------------------------</td>
<td>----------</td>
<td>---------------------</td>
</tr>
<tr>
<td>CHAR</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td></td>
<td>C</td>
<td></td>
<td>21</td>
<td></td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td></td>
<td>C</td>
<td></td>
<td>21</td>
<td></td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Numeric values in a column indicate the required output length.

X = Explicit length is required.

C = The default length is defined by the column.

**Character, graphic, and binary output**

The following table lists the conversions that NGT Unload supports to character, graphic, and binary output. The table that follows this one lists the default output lengths for these conversions.

**Table 12: Allowable data type conversions for character, graphic, and binary output fields**

<table>
<thead>
<tr>
<th>Output field</th>
<th>CHAR</th>
<th>VARCHAR</th>
<th>GRAPHIC</th>
<th>VARGRAPHIC</th>
<th>BINARY</th>
<th>VARBINARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input column</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 13: Default lengths for character, graphic, and binary output fields

<table>
<thead>
<tr>
<th>Output field</th>
<th>CHAR</th>
<th>VARCHAR</th>
<th>GRAPHIC</th>
<th>VARGRAPHIC</th>
<th>BINARY</th>
<th>VARBINARY</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input column</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECFLOAT</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>D</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td>X</td>
<td>D</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td>X</td>
<td>X</td>
<td>D</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The column values are:

- X = Allowable conversion
- D = Allowable conversion and indicates the default output data type, which is the data type of the column
## Date, time, and timestamp output

The following table lists the conversions that NGT Unload supports to date, time, and timestamp output. The table that follows this one lists the default output lengths for these conversions.

### Table 14: Allowable data type conversions for date, time, and timestamp output fields

<table>
<thead>
<tr>
<th>Output field</th>
<th>DATE EXTERNAL</th>
<th>TIME EXTERNAL</th>
<th>TIMESTAMP EXTERNAL</th>
<th>TIMESTAMP WITH TIME ZONE EXTERNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input column</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECFLOAT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Numeric values in a column indicate the required output length.

- **C** = The default length is defined by the column.
- **L** = The length is the value of the DATEFMT or TIMEFMT option, the local DATE or TIME length if your DSNHDEC format is LOCAL, or 10 (for DATE) and 8 (TIME).
- **P** = The default length is the precision + 2.
### Table 15: Default lengths for date, time, and timestamp output fields

<table>
<thead>
<tr>
<th>Output field</th>
<th>DATE EXTERNAL</th>
<th>TIME EXTERNAL</th>
<th>TIMESTAMP EXTERNAL</th>
<th>TIMESTAMP WITH TIME ZONE EXTERNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input column</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>X</td>
<td>X</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>D</td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XML</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The column values are:
- `X` = Allowable conversion
- `D` = Allowable conversion and indicates the default output data type, which is the data type of the column
<table>
<thead>
<tr>
<th>Output field</th>
<th>DATE EXTERNAL</th>
<th>TIME EXTERNAL</th>
<th>TIMESTAMP EXTERNAL</th>
<th>TIMESTAMP WITH TIME ZONE EXTERNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input column</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td>L</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>L</td>
<td>L</td>
<td>20</td>
<td>26</td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td>L</td>
<td>L</td>
<td>26</td>
<td>26</td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Numeric values in a column indicate the required output length.

C = The default length is defined by the column.

L = The length is the value of the DATEFMT or TIMEFMT option, the local DATE or TIME length if your DSNHDECP format is LOCAL, or 10 (for DATE) and 8 (TIME).

**LOB output**

The following table lists the conversions that NGT Unload supports to LOB output. The table that follows this one lists the default output lengths for these conversions.

**Table 16: Allowable data type conversions for LOB output fields**

<table>
<thead>
<tr>
<th>Output field</th>
<th>BLOB</th>
<th>CLOB</th>
<th>DBCLOB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input column</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output field</td>
<td>BLOB</td>
<td>CLOB</td>
<td>DBCLOB</td>
</tr>
<tr>
<td>--------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>Input column</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td>D</td>
</tr>
</tbody>
</table>

The column values are:

X = Allowable conversion
D = Allowable conversion and indicates the default output data type, which is the data type of the column

### Table 17: Default lengths for LOB output fields

<table>
<thead>
<tr>
<th>Output field</th>
<th>BLOB</th>
<th>CLOB</th>
<th>DBCLOB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input column</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECFLOAT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Additional data type considerations

This topic describes additional considerations for certain field data types.

**Scale specification**

With the exception of DECFLOAT fields, you can specify scale on any numeric (internal, EXTERNAL, or other format) fields with or without specifying a length. Specifying a scale multiplies or divides the number by a power of 10 or, for DECIMAL EXTERNAL, indicates where you want the decimal point. For example, INTEGER(,2) multiplies the number by 100, INTEGER(-3) divides the number by 1000, and DECIMAL EXTERNAL(5,2) gives you two digits to the right of the decimal point, no matter what the scale of the source value is.

If you do not specify scale, 0 is the default unless the column is DECIMAL and the field is DECIMAL(PACKED) (explicit or defaulted), or DECIMAL ZONED. In these cases, the scale defaults to the scale of the column.

**Restrictions**

The following restrictions apply to the scale specification:

- You cannot specify scale on a DECFLOAT field.
- With the exception of a decimal field, NGT Unload does not support the scale specification on an output field if you are converting from a DECFLOAT column.

**Date, time, or timestamp fields**

A date, time, or timestamp EXTERNAL field produces a character representation of the date, time, or timestamp value, producing an EUR, ISO, JIS, USA, or LOCAL format depending on the default date/time formats and lengths found in DB2 module DSNHDECP.
The default data format depends on the UNLOAD FORMAT option that is in effect.

### Variable fields

For VARCHAR, VARGRAPHIC, and VARBINARY data types, a two-byte length precedes the data. For VARCHAR and VARBINARY, the length is the number of single-byte characters. For VARGRAPHIC, the length is the number of double-byte characters. The length does not include the two length bytes.

### Data translation

Before writing the output record to the output data set, NGT Unload translates the data to the specified output encoding scheme. If you do not specify an encoding scheme, NGT Unload unloads the data in the EBCDIC encoding scheme.

NGT Unload does not support data translation when you specify DIRECT NO or DIRECT AUTO.

### Supported character conversions

The following table identifies the character conversions that NGT Unload supports for translation processing:

#### Table 18: Supported translations

<table>
<thead>
<tr>
<th>Target</th>
<th>ASCII SBCS</th>
<th>ASCII MIXED</th>
<th>ASCII DBCS</th>
<th>EBCDIC SBCS</th>
<th>EBCDIC MIXED</th>
<th>EBCDIC DBCS</th>
<th>UNICODE SBCS</th>
<th>UNICODE MIXED</th>
<th>UNICODE DBCS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII SBCS</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASCII MIXED</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASCII DBCS</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EBCDIC SBCS</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EBCDIC MIXED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EBCDIC DBCS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>UNICODE SBCS</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>UNICODE MIXED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UNICODE DBCS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This table indicates the supported translations with the following identifiers:

X: allowable translation for either DIRECT YES or DIRECT NO
Y: allowable translation for DIRECT YES only

### Expansion and contraction

The translation process might cause the length of the resultant data to change based on the CCSID type that you specified. To determine the resultant length of the data based on each CCSID conversion, see the IBM DB2 SQL Reference.

### Mixed data

Translation is not supported from Unicode MBCS (UTF-8) to EBCDIC or ASCII MIXED on a DB2 subsystem that is defined as MIXED=NO, because the target CCSID value is undefined. However, NGT Unload and other SQL applications such as SPUFI and DSNTIAUL, extract MIXED columns as SBCS data. During the translation process, character substitutions might occur between the source CCSID and target CCSID; a loss of data integrity can result, due to SUBBYTE characters in the translated data.

**WARNING**

If you specify the NOSUBS option with DIRECT NO, NGT Unload ignores the NOSUBS option for translations that DB2 handles. In cases in which NGT Unload handles row-level data translation (such as INTO field specifications or reformatting of DATE, TIME, or TIMESTAMP columns), specifying NOSUBS might result in NGT Unload discarding rows or terminating.

### Command constants

For comparisons between command constants and row data, NGT Unload must translate certain command constants from EBCDIC (using the DB2 installation default EBCDIC SBCS CCSID) to the encoding scheme of the table. NGT Unload translates the following character constants for comparison:

- Predicate block constants
- LIKE constants
- IN constants
NGT Unload translates output data from EBCDIC (using the DB2 installation default EBCDIC SBCS CCSID) to the output encoding scheme. NGT Unload translates the following character constants for output:

- AUTOTAG values
- SELECT constants
- IF VALUE constants
- NULLCHAR

### Order of data type conversion and data translation

NGT Unload supports data type conversions from one encoding scheme to another. The following tables detail when translation occurs with respect to the data type conversion that NGT Unload is performing.

<table>
<thead>
<tr>
<th>Output field</th>
<th>SMALLINT</th>
<th>INTEGER</th>
<th>BIGINT</th>
<th>DECIMAL</th>
<th>DECIMAL ZONED</th>
<th>FLOAT</th>
<th>All numeric external output except DECFLOAT</th>
<th>DECFLOAT</th>
<th>DECFLOAT EXTERNAL</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>
<td>cpt 2</td>
<td>x</td>
<td>cpt 2</td>
</tr>
<tr>
<td>INTEGER</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>cpt 2</td>
<td>x</td>
<td>cpt 2</td>
</tr>
<tr>
<td>BIGINT</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>cpt 2</td>
<td>x</td>
<td>cpt 2</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>cpt 2</td>
<td>x</td>
<td>cpt 2</td>
</tr>
<tr>
<td>FLOAT</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>cpt 2</td>
<td>x</td>
<td>cpt 2</td>
</tr>
<tr>
<td>CHAR</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
<td>tc1</td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The character combinations in this table indicate the order in which data type conversions (c), data translations (t), and character padding (p) occur. The numbers (1,2,3) represent the type of data translation that occurs when converting from one data type to another.

- **x** = Support conversion, no translation required
- **c** = Convert
- **t** = Translate
- **p** = Pad with blank

1 = Translate input CCSID to EBCDIC SBCS (system default)
2 = Translate EBCDIC SBCS (system default) to output CCSID
3 = Translate input CCSID to output SBCS

### Table 20: Conversion and translation processing for character, graphic, and binary output

<table>
<thead>
<tr>
<th>Output field</th>
<th>CHARACTER</th>
<th>VARCHAR</th>
<th>GRAPHIC</th>
<th>VARGRAPHIC</th>
<th>BINARY</th>
<th>VARBINARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input column</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td>cpt 2</td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>cpt 2</td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td>cpt 2</td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td>cpt 2</td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td>cpt 2</td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>cpt 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td>cpt 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>cpt 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The character combinations in this table indicate the order in which data type conversions (c), data translations (t), and character padding (p) occur. The numbers (1,2,3) represent the type of data translation that occurs when converting from one data type to another.

- x = Support conversion, no translation required
- c = Convert
- t = Translate
- p = Pad with blank
- 1 = Translate input CCSID to EBCDIC SBCS (system default)
- 2 = Translate EBCDIC SBCS (system default) to output CCSID
- 3 = Translate input CCSID to output SBCS

**Table 21: Conversion and translation processing for date, time, and timestamp output**

<table>
<thead>
<tr>
<th>Output field</th>
<th>CHARACTER</th>
<th>VARCHAR</th>
<th>GRAPHIC</th>
<th>VARGRAPHIC</th>
<th>BINARY</th>
<th>VARBINARY</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input column</strong></td>
<td><strong>DATE EXTERNAL</strong></td>
<td><strong>TIME EXTERNAL</strong></td>
<td><strong>TIMESTAMP EXTERNAL</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td>ctp2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td>ctp2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NGT Unload option descriptions—UNLOAD syntax
The character combinations in this table indicate the order in which data type conversions (c), data translations (t), and character padding (p) occur. The numbers (1,2,3) represent the type of data translation that occurs when converting from one data type to another.

- x = Support conversion, no translation required
- c = Convert
- t = Translate
- p = Pad with blank
- 1 = Translate input CCSID to EBCDIC SBCS (system default)
- 2 = Translate EBCDIC SBCS (system default) to output CCSID
- 3 = Translate input CCSID to output SBCS

### Table 22: Conversion and translation processing for LOB output

<table>
<thead>
<tr>
<th>Output field</th>
<th>BLOB</th>
<th>CLOB</th>
<th>DBCLOB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input column</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIGINT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOAT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINARY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARBINARY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Output field

<table>
<thead>
<tr>
<th>Input column</th>
<th>BLOB</th>
<th>CLOB</th>
<th>DBCLOB</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIMESTAMP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOB</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>DBCLOB</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>DECIMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The character combinations in this table indicate the order in which data type conversions (c), data translations (t), and character padding (p) occur. The numbers (1,2,3) represent the type of data translation that occurs when converting from one data type to another.

- x = Support conversion, no translation required
- c = Convert
- t = Translate
- p = Pad with blank
- 1 = Translate input CCSID to EBCDIC SBCS (system default)
- 2 = Translate EBCDIC SBCS (system default) to output CCSID
- 3 = Translate input CCSID to output SBCS

### UNLOAD syntax that is no longer valid

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

<table>
<thead>
<tr>
<th>Invalid option</th>
<th>NGT Unload action</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVE</td>
<td>Ignores</td>
</tr>
<tr>
<td>ANALYZE</td>
<td>Ignores</td>
</tr>
<tr>
<td>CENTURY</td>
<td>Terminates</td>
</tr>
<tr>
<td>CLONE</td>
<td>Terminates</td>
</tr>
<tr>
<td>CURRENTDEGREE</td>
<td>Ignores</td>
</tr>
<tr>
<td>DDLDDN</td>
<td>Terminates</td>
</tr>
<tr>
<td>DECFLOAT_ROUNDMODE</td>
<td>Terminates</td>
</tr>
<tr>
<td>DSPLOCKS</td>
<td>Ignores</td>
</tr>
<tr>
<td>ENUMROWS</td>
<td>Ignores</td>
</tr>
<tr>
<td>FORCE - all keywords except NONE</td>
<td>Terminates</td>
</tr>
<tr>
<td>Invalid option</td>
<td>NGT Unload action</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>FORMAT - all keywords except EXTERNAL, DSNTIAUL, and UNLOAD</td>
<td>Terminates</td>
</tr>
<tr>
<td>IF ERROR (on the field specification)</td>
<td>Terminates</td>
</tr>
<tr>
<td>IMPLICIT_TZ</td>
<td>Terminates</td>
</tr>
<tr>
<td>INTERVAL</td>
<td>Terminates</td>
</tr>
<tr>
<td>MAXBLKSIZE</td>
<td>Terminates</td>
</tr>
<tr>
<td>MAXCONNECT</td>
<td>Ignores</td>
</tr>
<tr>
<td>MAXSORTS</td>
<td>Ignores</td>
</tr>
<tr>
<td>ON FAILURE with RETCODE</td>
<td>Ignores</td>
</tr>
<tr>
<td>ON MESSAGE</td>
<td>Terminates</td>
</tr>
<tr>
<td>RECFM</td>
<td>Terminates</td>
</tr>
<tr>
<td>SET CURRENT QUERY ACCELERATION ALL</td>
<td>Terminates</td>
</tr>
<tr>
<td>SORTDEVT</td>
<td>Ignores</td>
</tr>
<tr>
<td>SORTNUM</td>
<td>Ignores</td>
</tr>
<tr>
<td>SYNC</td>
<td>Ignores</td>
</tr>
<tr>
<td>UNLOADEXIT</td>
<td>Terminates</td>
</tr>
<tr>
<td>USELRECL</td>
<td>Terminates</td>
</tr>
<tr>
<td>WTOMSG</td>
<td>Ignores</td>
</tr>
<tr>
<td>XBMID</td>
<td>Ignores</td>
</tr>
<tr>
<td>ZIIP</td>
<td>Ignores</td>
</tr>
</tbody>
</table>

NGT Unload option descriptions—UNLD syntax

The following topics describe the options that are available with the UNLD command.

Future enhancements will be made to the UNLOAD command only. The UNLD command will be deprecated in a future release.
**APPEND**

Use this option to instruct NGT Unload to append data to the end of the UNLOAD data set.

You can achieve the same results by specifying `DISP=MOD` on the UNLD DD statement.

**ASCII**

Use this option to tell NGT Unload to use the DB2 installation default (DSNHDECP) CCSIDs to encode the unloaded data in ASCII.

If you do not specify ASCII, UNICODE, or EBCDIC on the unload statement, the default is based on the encoding scheme of the unloaded table, as follows:

- ASCII encodes data using EBCDIC CCSID defined to DB2.
- UNICODE encodes data using EBCDIC CCSID defined to DB2.
- EBCDIC encodes data using EBCDIC CCSID defined to DB2 if you specify MODE(AUTO), but does not encode data if you specify MODE(DIRECT).

**Note**

Do not use MODE(DB2SQL) or MODE(AUTO) with this option.

For more information about page encoding, see “Page encoding (UNLD command)” on page 27.

**BLANKPAD**

Use the BLANKPAD option to pad data in VARCHAR columns with blanks instead of low values.

The BLANKPAD option overrides the +BLANKPAD parameter value.

**Note**

Do not use this option to prevent NGT Unload from padding VARCHAR columns. Instead, use the NOPAD option. For more information, see “PAD or NOPAD” on page 147.
**BYPART**

BYPART tells NGT Unload to unload individually to separate files the partitions that you specify with the PARTS option. If you do not specify the PARTS option with BYPART, NGT Unload unloads the total number of partitions in the table space individually to separate files.

The BYPART option applies only when the target table to be unloaded is partitioned and dynamic allocation is in use. If not, NGT Unload ignores this option.

You cannot use the BYPART option in combination with the following options:

- COPYDDN
- COPYDSN
- SYSCOPY
- MODE(AUTO)
- MODE(DB2SQL)

**Example**

The following statement:

```
UNLD BYPART PARTS 1,5:7,20 FROM TABLE TABLE1
```

achieves the same effect as the following statement:

```
UNLD PARTS  1   FROM TABLE TABLE1
UNLD PARTS  5   FROM TABLE TABLE1
UNLD PARTS  6   FROM TABLE TABLE1
UNLD PARTS  7   FROM TABLE TABLE1
UNLD PARTS  20  FROM TABLE TABLE1
```

However, if the original UNLD BYPART option uses a WHERE clause that causes certain parts to be filtered, NGT Unload individually unloads only the applicable parts.

**CCSID**

Use this option to specify the coded character set ID (CCSID) that NGT Unload uses to encode unloaded data. The values that you specify must be in the order shown in the syntax diagram.

NGT Unload extracts the source CCSIDs from the table definition in the DB2 catalog and uses z/OS Conversion Services to carry out the text translation from the source CCSIDs to the target CCSIDs.

For more information about page encoding, see “Page encoding (UNLD command)” on page 27.
**WARNING**

Do not use MODE(DB2SQL) or MODE(AUTO) with this keyword.

---

**COPYDDN**

Use this option to specify a DD statement name that points to an image copy data set that contains table data to be unloaded.

The data set organization rules for this option are the same as those for COPYDSN (see COPYDSN on page 132). However, with the COPYDDN option, you can concatenate data sets that have the same organization.

NGT Unload concatenates VSAM data sets and dynamically reallocates with DISP=OLD unless VOL=PRIVATE is specified on the DD statement.

As an alternative to specifying an image copy data set name in your DD statement, you can use the OUTPUT command. You must place the OUTPUT command before the UNLD or UNLOAD statement in your SYSIN. For more information about this command, see the *BMC Next Generation Technology General User Guide*.

**Restrictions**

The following restrictions apply to this option:

- When unloading from an image copy, you cannot specify multiple partitions with the PARTS option.
- Do not use MODE(DB2SQL) or MODE(AUTO) with this option.

---

**COPYDSN**

Use this option to specify the name of an image copy data set.

The data set name cannot be a PDS or member of a PDS. You can use relative GDG members in the data set name (for example, (0) or (-1)).

The data set can be on tape or disk, but it must be cataloged. The data set must be either sequential or a VSAM linear data set. During the unload process, NGT Unload dynamically allocates the data set with a DISP=SHR, if sequential, or DISP=OLD, if VSAM.

**Restrictions**
The following restrictions apply to this option:

- When unloading from an image copy, you cannot specify multiple partitions with the PARTS option.
- Do not use MODE(DB2SQL) or MODE(AUTO) with this option.

**DATA**

This keyword is optional.

**DATEFMT**

Use this option to override the ULDPARMS +DATEFMT parameter for the current unload statement only.

DATEFMT enables NGT Unload to generate a variety of DATE formats other than the standard DB2 formats.

Specify a string of up to 40 characters as an argument for DATEFMT. NGT Unload replaces character strings to generate the actual date in the formats shown in the following table. All other characters remain unchanged.

<table>
<thead>
<tr>
<th>Character string</th>
<th>Date format</th>
</tr>
</thead>
<tbody>
<tr>
<td>YYYY</td>
<td>4-digit year</td>
</tr>
<tr>
<td>YY</td>
<td>2-digit year</td>
</tr>
<tr>
<td>MM</td>
<td>2-digit month</td>
</tr>
<tr>
<td>DD</td>
<td>2-digit day of the month</td>
</tr>
</tbody>
</table>

**Example**

If today is July 17, 1998, the character string YYYY.MM.DD results in the date format 1998.07.17.

You cannot code more than one instance of a year, day, or month element in a DATEFMT string. For example, the string YYYY/YY/MM/DD is invalid because the year is repeated.
DIGITS

Use the DIGITS option to enforce compatibility between certain DB2 SQL functions and NGT SQL functions.

For more information about the DIGITS function, see the NGT SQL language reference information in the *BMC Next Generation Technology General User Guide*.

EBCDIC

Use this option to tell NGT Unload to use the DB2 installation default (DSNHDECP) CCSIDs to encode the unloaded data in EBCDIC.

If you do not specify ASCII, UNICODE, or EBCDIC on the unload statement, the default is based on the encoding scheme of the unloaded table, as follows:

- ASCII encodes data using EBCDIC CCSID defined to DB2.
- UNICODE encodes data using EBCDIC CCSID defined to DB2.
- EBCDIC encodes data using EBCDIC CCSID defined to DB2 if you specify MODE(AUTO), but does not encode data if you specify MODE(DIRECT).

**Note**
Do not use MODE(DB2SQL) with this option.

For more information about page encoding, see “Page encoding (UNLD command)” on page 27.

EXCLUDE PARTS

Use this option with the UNLD command to tell NGT Unload to exclude specified partitions from unload processing.

You can specify a list of partitions, a range of partitions, or both. To specify a range of partitions, use the format \(x:y\), where \(y\) is greater than \(x\).

The list of partitions or ranges does not have to be in order. However, the list cannot contain duplicate partitions. For example, the following list is valid:

```
EXCLUDE PARTS (10,2:3,6)
```

But the following list is invalid:

```
EXCLUDE PARTS (10,2:3,2:6)
```
FORMAT

Use this option to identify the desired format of the output file. This option is optional.

The following table lists valid formats and their valid abbreviations.

Table 24: Valid formats and abbreviations

<table>
<thead>
<tr>
<th>Format</th>
<th>Abbreviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSNTIAUL</td>
<td>DSN or IBM</td>
</tr>
<tr>
<td>EXTERNAL</td>
<td>EXT</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>UNL</td>
</tr>
<tr>
<td>DELIMITED</td>
<td>DEL</td>
</tr>
<tr>
<td>ASCIIDEL</td>
<td>ASC</td>
</tr>
</tbody>
</table>

For more information about these formats, see “Output file formats” on page 167.

**DSNTIAUL and EXTERNAL**

You can use the NULLCHAR keyword with the FORMAT DSNTIAUL and FORMAT EXTERNAL options. For the NULLCHAR syntax, see the null specification diagram in NGT Unload syntax diagrams on page 37.

When you download a nullable field from the table, NGT Unload places a marker in the unloaded data trailing the field, leading it, or in front of the unloaded record. The marker may have from one to four characters.

With the UPFRONT specification, NGT Unload places all nullable column markers in the entire record at the beginning of the record in an array format. For example, if four nullable columns are in the record and the null marker is ?, the first and third are nulls, but the second and fourth are not. Therefore, there are four characters at the beginning of the record corresponding to the four nullable columns. These characters have the values in hexadecimal X'6F006F00', where X'6F' is ?. NGT Unload uses this hexadecimal when converting from CA unload. The generated load statement has a NULLIF clause with correct position values in the null markers array.

The NULLCHAR specification must be coded immediately after FORMAT DSNTIAUL or FORMAT EXTERNAL.

The default value for string is a question mark ("?"). The default position is TRAILING.
The field content when its value is null is binary zeros. Conversely, when the nullable field is not null, the null marker is set to binary zeros.

**UNLOAD**

The following considerations apply to FORMAT UNLOAD:

- Do not use MODE(DB2SQL) or MODE(AUTO) with FORMAT UNLOAD.
- If you specify a SELECT statement that uses a subset of columns in the table, you cannot reload the results into any table. Use this type of SELECT statement with FORMAT UNLOAD only to produce an output file that you can then reference in a WHERE or WHEN clause of a discard specification in NGT Reorg, as shown in the following example:

```
DISCARD FROM TABLE tableName WHERE colName IN
(FILE(fileName),FORMAT(UNLOAD))
```

**DELIMITED**

You can use the following additional keywords with the FORMAT DELIMITED option.

**COLDEL**

Specifies the column delimiter to use in the unload file. The default value is a comma. For ASCII and UTF-8 data, this is X'2C'; for EBCDIC data it is X'6B'.

**CHARDEL**

Specifies the character string delimiter to use in the unload file. The default value is a double quotation mark. For ASCII and UTF-8 data this is X'22'; for EBCDIC data, it is X'3F'.

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

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

**DECPT**

Specifies the decimal point character to use in the unload file. The default value is a period. For ASCII and UTF-8, this is X'2E'. The only allowed
delimiters for DECPT are the period (.) or the comma (,), or their equivalents in ASCII or UNICODE.

**ASCIIDEL**

You can use the following additional keywords with the FORMAT ASCIIDEL option.

**FIELDSEP**

Overrides the ULDPARMS +FIELDSEP parameter for an individual unload statement.

You can specify any single character.

**CHARFLDDLDM**

Overrides the ULDPARMS +CHARFLDDLDM parameter for an individual unload statement.

The specification allows a single character to the left of the field and a single character to the right of the field. If the AND is omitted, then NGT Unload uses the same character for left and right.

**NULLSTRING**

Sets a value for a nullable field that is NULL.

The maximum length of the null string is four and it defaults to none (for example, a string of length 0).

**FROM [TABLE]**

Use the FROM option to specify the table from which NGT Unload should unload data records.

You can specify a table name, view name, synonym, or alias.
The TABLE keyword is optional. You cannot specify the TABLE keyword when you specify MODE(DB2SQL).

**AS [TABLE]**

You can optionally specify the AS keyword with the FROM option to specify a table name to be used in the INTO clause of the generated LOAD statement. If you do not specify AS, NGT Unload uses the table that you specified with the FROM option. You can specify only a table name with AS, and the keyword TABLE is optional.

---

**Example**

FROM oldTableName AS newTableName results in the following INTO clause of the generated LOAD statement:

```sql
INTO newTableName
```

Whereas FROM oldTableName results in the following INTO clause of the generated LOAD statement:

```sql
INTO oldTableName
```

---

**LOADSTMT**

Use this option to customize LOAD statements generated by NGT Unload.

This character string should not exceed 77 characters in length. However, you can use multiple strings (delimited by single quotation marks) for groups of options that exceed 77 characters. The example in this topic illustrates this scenario.

NGT Unload inserts all options, without any changes, between the LOAD statement and the INTO TABLE statement.

---

**Example**

You can code multiple options in a LOAD statement, as in the following example:

```sql
LOADSTMT 'ENFORCE CONSTRAINTS DISCARDDN(TB1DISC) DISCARDS 100' 'COPYDDN(PCOPY1) RECOVERYDDN(BCOPY1)'
```

This example could produce the following SYSPUNCH:

```sql
LOAD DATA INDDN(SYSREC00)
ENFORCE CONSTRAINTS DISCARDDN(TB1DISC) DISCARDS 100
COPYDDN(PCOPY1) RECOVERYDDN(BCOPY1)
INTO TABLE DB1.TB1
( "COL01" POSITION(1:11) INTEGER EXTERNAL(11)
, "COL02" POSITION(12:20) CHAR(9)
, ....
, ....
, ....)
```
Alternatively, you can specify a data set that contains these strings.

The following considerations apply to the LOADSTMT option:

- You can use this option to customize a LOAD statement for all formats except FORMAT ASCIIDEL and FORMAT UNLOAD.
- Do not specify multiple LOADSTMT options to group multiple statements. Doing so causes a syntax failure.

**MAXERR**

Use this option to specify the maximum number of allowed records in error.

If this limit is reached, the unload terminates and issues message NGTN071 or NGTN072.

**MAXLRECL**

Use this option to specify a maximum logical record length of the UNLD data set. This value cannot exceed 32756.

The default value is based on the FORMAT and the table column structure. When you specify this option, NGT Unload sets a RECFM of VB for the data sets, unless it conflicts with existing attributes. If you are planning to append data to the UNLD data set, specify a number that is large enough to accommodate future unloads.

**MAXROWS**

Use this option to limit the number of rows that NGT Unload unloads from a table.

When NGT Unload reaches the specified integer or reaches the end of the file (whichever occurs first), NGT Unload terminates processing. If you specify a value for MAXROWS that is greater than the number of table rows, NGT Unload unloads all table rows. To unload all rows without limit, specify **MAXROWS 0** or omit this option.

The FILSZ value passed to SORT comes from the MAXROWS option value (if it is used). Otherwise, the FILSZ value uses a value of 100,000.

For an example of the use of the MAXROWS option, see “Example: unloading a specific number of rows” on page 165.
MINROWS

Use this option to specify a minimum number of rows to unload. This option tells NGT Unload to issue a warning (RC=4) if the number of unloaded rows is less than the specified number.

You can specify any number greater than or equal to 0.

If you specify 0, then NGT Unload uses the global +MINROWS value specified in the ULDPARMS DD statement.

MODE

Use the MODE option to tell NGT Unload whether to execute the SELECT clause under the DB2 full-function SQL.

You can specify one of the following keywords for the MODE option:

- DIRECT
- DB2SQL
- AUTO

You should place the MODE option after the UNLD DATA options, but before the SELECT statement.

**Note**

If you specify MODE(DB2SQL) or MODE(AUTO), you must specify a SELECT statement.

Using any of the following options with MODE(DB2SQL) or MODE(AUTO) results in a syntax error:

- COPYDDN
- COPYDSN
- LOADINDD
- SYSCOPY
- PART
- PARTS
- CCSID
- ASCII
- FORMAT UNLOAD
DIRECT

Use MODE(DIRECT), the default, to tell NGT Unload to operate under the NGT direct method.

Unless you have complex data selections that require MODE(DB2SQL), such as joins or column functions, BMC recommends that you use MODE(DIRECT). This mode is the most efficient mode in terms of elapsed time and CPU time.

In this mode, you can code the SELECT, WHERE, FROM, or ORDER BY clauses anywhere and in any order within the scope of an NGT Unload statement. For example:

```
UNLD MODE(DIRECT)
ORDER BY 1 FORMAT DSN
WHERE C1=C2
SELECT * PAD FROM TBL1
```

DB2SQL

Use MODE(DB2SQL) to tell NGT Unload to extract the SQL statement from an NGT Unload statement and process it using the IBM DB2 SQL processor.

The following considerations apply to MODE(DB2SQL):

- NGT Unload formats the output to NGT Unload formats, including DSNTIAUL, EXTERNAL, ASCIIDEL, and DELIMITED.

- NGT Unload supports all data types, including large objects (LOBs). However, NGT Unload truncates LOB and LONG VARCHAR columns to fit the maximum record length of 32,752 bytes. If there is more than one LOB column, the truncation occurs evenly across all LOB columns. You cannot specify SPANNED YES with MODE(DB2SQL).

- If you use dynamic allocation under MODE(DB2SQL), NGT Unload uses the MAXROWS value to estimate the output data set primary space allocation. If you do not specify the MAXROWS value, NGT Unload uses the real-time statistics (RTS) table to determine the number of rows per partition.
For all applicable formats, NGT Unload reformats the DATE, TIME, and TIMESTAMP fields based on the specifications of the options shown in the following table:

Table 25: Options that affect the format of date/time fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE</td>
<td>DATEFMT</td>
</tr>
<tr>
<td>TIME</td>
<td>TIMEFMT</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>TSFMT</td>
</tr>
</tbody>
</table>

**Note**
NGT Unload does not use the global formatting options specified in the ULDPARMS DD statement to reformat DATE, TIME, or TIMESTAMP fields.

You cannot use the TABLE keyword with the FROM option when you specify MODE(DB2SQL).

**Example**
The following is an example of an UNLD statement that uses MODE(DB2SQL):

```
UNLD DATA FORMAT(EXTERNAL) OUTDDN(OUTDSET) MODE(DB2SQL) SELECT CHAR(DT_EFF, USA), CHAR(DT_STAT, ISO), CHAR(DT_TERM, EUR) FROM CORP.AR_RECORDS
```

**AUTO**

Use MODE(AUTO) to tell NGT Unload to automatically switch to MODE(DB2SQL) when encountering an SQL statement that MODE(DIRECT) does not support. The following conditions cause NGT Unload to automatically switch to MODE(DB2SQL):

- The SELECT or WHERE clause contains unsupported syntax
- The SELECT clause results in an incompatible data type or nullability attribute with MODE(DB2SQL)
- A complex view definition exists.
- An output row size exceeding 32,756 bytes exists
- An ORDER BY key length exceeding 4,090 bytes exists
- A complex expression exists

If it does not encounter any of these conditions, NGT Unload processes the statement using the NGT direct method.
When NGT Unload selects the NGT direct method under MODE(AUTO), NGT Unload changes the following processing options to maintain compatibility with DB2SQL mode:

- +DIGITS(IBM) is enforced. This forces the NGT SQL functions DIGITS and CHAR to produce output compatible with DB2 SQL functions DIGITS and CHAR.

- NGT Unload always includes a selected ROWID column in the data, whether defined with the GENERATED ALWAYS attribute or by default.

When you perform an unload under DB2SQL, NGT Unload issues the following message: NGTN049 UNLOAD VIA DB2SQL

**WARNING**

For both modes (DB2SQL or AUTO), do not place the SELECT clause in parentheses and ensure that no NGT Unload options follow it. That is, all NGT Unload keywords must precede the SELECT keyword.

**MRF**

Use this keyword to tell NGT Unload to use a multi-row fetch for a specified number of rows. This can significantly improve performance by reducing CPU time. Use this keyword only when you have specified MODE(DB2SQL) or MODE(AUTO). You can specify an integer between 0 and 32767. NGT Unload is shipped without a value for this option. BMC recommends that you specify a value of 100.

**Data compatibility between MODE(DIRECT) and MODE(DB2SQL)**

Certain specifications produce different results in MODE(DIRECT) than in MODE(DB2SQL). Because MODE(AUTO) might switch to MODE(DB2SQL) when encountering an unsupported condition, this section provides suggestions for ensuring that you get consistent results.

**Character string constants**

In MODE(DB2SQL), NGT Unload treats a character string constant as a VARCHAR. In MODE(DIRECT), a character string constant results in a fixed-length character string.

In the following example, MODE(AUTO) in the first UNLD statement has switched to MODE(DB2SQL), whereas the absence of the MODE option in the second statement tells NGT Unload to use MODE(DIRECT).
Example

The following statement:

```
UNLD MODE(AUTO) SELECT 'ABC' AS FIELD1 FROM SYSIBM.SYSDUMMY1
```

results in the following field specification:

```
(FIELD1 POSITION(1) VARCHAR)
```

However, the following statement:

```
UNLD SELECT 'ABC' AS FIELD1 FROM SYSIBM.SYSDUMMY1
```

results in the following field specification:

```
(FIELD1 POSITION(1) CHAR(3))
```

To ensure that either mode produces the same results, specify the following UNLD statement:

```
UNLD MODE(AUTO) SELECT CHAR('ABC',3) AS FIELD1 FROM SYSIBM.SYSDUMMY1
```

Arithmetic on non-nullable fields

Nullability rules for MODE(DIRECT) are the same as for MODE(DB2SQL) except when performing arithmetic on two non-nullable fields (fields that are defined as NOT NULL). In MODE(DB2SQL), the result is nullable, whereas MODE(DIRECT), results in a NOT NULL attribute.

In the following example, MODE(AUTO) in the first UNLD statement has switched to MODE(DB2SQL), whereas the absence of the MODE option in the second statement tells NGT Unload to use MODE(DIRECT).

Example

Assuming that F1 and R2 are integer fields defined as NOT NULL, the following UNLD statement:

```
UNLD MODE(AUTO) SELECT F1 + F2 AS FIELD1 FROM TBL1
```

results in the following field specification:

```
(FIELD1 POSITION(1) INTEGER NULLIF(5) = '?')
```

However, the following statement:

```
UNLD SELECT F1 + F2 AS FIELD1 FROM TBL1
```

results in the following field specification:

```
(FIELD1 POSITION(1) INTEGER)
```

To ensure that either mode produces the same results, specify one of the following UNLD statements:

- UNLD MODE(DB2SQL) SELECT IFNULL(F1+F2,0)
- UNLD MODE(AUTO) SELECT IFNULL(CASE WHEN INT(1) = 1 THEN F1 + F2 ELSE NULL END,0)
The NOFILTER option tells NGT Unload to unload all partitions, regardless of any WHERE clause on the UNLD command.

If you do not specify NOFILTER, NGT Unload automatically skips partitions that do not satisfy the criteria for the WHERE clause on your UNLD command. For more information about the WHERE clause, see the BMC Next Generation Technology General User Guide.

Use this option to look for a different table OBID value when unloading from an image copy.

Without this option, NGT Unload assumes that the input data has the same OBID as the one defined in SYSIBM.SYSTABLES for the table named in the FROM TABLE clause. If this is not what you intend, use the OBID option to specify a different one. NGT Unload does not validate the OBID value that you supply. Supply the value in integer format, not hexadecimal format. The parentheses are optional.

For an example that uses this option, see “Example: unloading from a copy” on page 166.

Use the ORDER BY option to tell NGT Unload how to order the unloaded rows.

Use one of the following specifications to indicate the order for the unloaded rows:

.columnName or columnNumber

This specification tells NGT Unload to order data based on the indicated columns before unloading the table. Specify one of the following keywords with the column name or number:

- ASC tells NGT Unload to sort the data in ascending order.
- DESC tells NGT Unload to sort the data in descending order.
- RANDOM tells NGT Unload to put the data in random order.

You do not need to include these columns in your select list.
Instead of specifying column names, you can specify column numbers based on their position in the select list.

**SQLExpression**

Use a valid SQL expression to tell NGT Unload to select for unload only the rows that satisfy the criteria described in the expression. You cannot include a constant in this expression. Specify one of the following keywords with the expression:

- **ASC** tells NGT Unload to sort in ascending order the data that satisfies the expression.
- **DESC** tells NGT Unload to sort in descending order the data that satisfies the expression.
- **RANDOM** tells NGT Unload to put in random order the data that satisfies the expression.

**CLUSTER**

This keyword tells NGT Unload to unload the rows in clustering index order. In the absence of a clustering index, NGT Unload ignores this option.

**OUTDDN**

This option names the DD statement that contains the unloaded data.

If you do not specify this option and do not use dynamic allocation, NGT Unload attempts to open and write to the default DD name of SYSREC00. You must allocate the specified DD name or use one of the following methods to dynamically allocate the data set. If you dynamically allocate the data set, the value of the `+OVERRIDEOUTPUT` parameter determines the method that NGT Unload uses.

- Providing the data set name in the XULDDYNM automation control point.
- Providing the data set name in an OUTPUT command in your unload SYSIN.

You must place the OUTPUT command before the UNLD or UNLOAD statement in your SYSIN.

For more information about XULDDYNM, 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*.
Note
If you specify OUTDDN with a DD and data set (for example, if SYSREC DD points to a named sequential data set), then NGT Unload performs an unload in the master job, rather than in any server job.

**PAD or NOPAD**

Use PAD to tell NGT Unload to pad a VARCHAR column to its full length. Use NOPAD to tell NGT Unload to unload the column as is, without any padding.

You cannot specify PAD with the ASCIIDEL format.

If you do not specify PAD or NOPAD, the defaults are as follows:

- For the DSNTIAUL and EXTERNAL formats, the default is PAD unless you specify `RECFM=V` on the OUTDDN DD statement.
- For the UNLOAD format, the default is PAD.

**PARTS or LPARTS**

Use these options to specify a list of partitions. PARTS indicates that the listed partitions are physical partitions. LPARTS indicates that the partitions are logical partitions. You cannot specify both PARTS and LPARTS.

NGT Unload unloads all listed partitions. Partition lists consist of numbers that represent a single partition number separated by commas, or ranges that represent a range of numbers separated by a colon.

**Example**

The specification `1,5,7:9,20:50` tells NGT Unload to unload partitions 1, 5, 7, 8, and 9, and all partitions between 20 and 50.

If a partition list includes the same partition more than once, an error occurs.

**Example**

The partition list `1,5,2:9` is invalid because partition 5 is specified twice, once explicitly and a second time as part of a range.

To include blanks in a partition list, enclose the entire list in parentheses.
Note
Do not use MODE(DB2SQL) or MODE(AUTO) with this option.

ROWEXIT

Use this keyword to invoke the ROWEXIT module every time NGT Unload unloads a row from a DB2 table.

The ROWEXIT module can change, leave unchanged, or ignore the data in each row. For more information about the ROWEXIT module, see ROWEXIT module on page 179.

The ROWEXIT keyword value must specify the name of the user exit to be called from NGT Unload. This user exit must exist in the load library concatenation referenced in the utility JCL. Otherwise, the job fails with error message NGTN038, indicating that the user exit is not available for processing.

Example

UNLD DATA FORMAT(DSNTIAUL) ROWEXIT(MODNAME)
FROM TABLE NGT.UNLOAD_TEST

In this unload command, MODNAME represents the 8-character user exit module name in the load library concatenation.

SELECT

Use the SELECT option to tell NGT Unload which columns to unload. You can specify only one SELECT option on your UNLD command.

For MODE(DB2SQL) and MODE(AUTO), the following requirements apply to SELECT:

- You must specify a SELECT clause (the clause that includes SELECT, FROM, WHERE, and ORDER BY) and it must be at the end of your UNLD command.
- You must specify the SELECT keyword and it must be the first keyword in the clause.

Use one of the following methods to specify the columns to unload:
* (asterisk)

Specifying * tells NGT Unload to unload all columns from the specified table except the following columns:
- Hidden columns
- ROWID columns

This value is the default.

columnName

Specifying one or more column names tells NGT Unload to unload the indicated columns from the specified table.

eexpression

Specifying an expression tells NGT Unload to unload a subset of the columns from the specified table, or to unload the columns in an order that differs from original order of the specified table.

You can use expressions to refer to an operation on one or more columns.

For more information about using expressions in NGT utility products, see the *BMC Next Generation Technology General User Guide*. 

**SHRLEVEL CHANGE**

Use SHRLEVEL CHANGE to create an unload file while concurrent changes are taking place.

The resulting unload is not a consistent point-in-time (PIT). If you do not specify SHRLEVEL CHANGE, NGT Unload creates a consistent PIT online unload with concurrent changes. Specify SHRLEVEL CHANGE only when obtaining a drain on writers for the object is an issue.

**SPANNED YES**

The SPANNED YES option enables you to unload the LOB auxiliary table and the base table in a single unload job. NGT Unload unloads the data to a variable-block spanned (VBS) data set.

The following considerations apply to this option:
- You do not need an NGT LOBMaster license to use this feature.
If you do not specify SPANNED YES, NGT Unload unloads only the base table—even if your unload selection includes a LOB column.

If you also specify the OUTPUT command for your output data set, SPANNED YES takes precedence over the OUTPUT command.

NGT Unload ignores any translation specification and unloads the data using the encoding scheme of the current table. The LOAD statement that NGT Unload generates includes the CCSID option with the current encoding scheme.

**WARNING**
Do not use MODE(DB2SQL) or MODE(AUTO) with this option.

**STMTDDN**

Use this option to name the DD statement to which NGT Unload writes the generated LOAD statement.

If the DD name represents a sequential data set, then NGT Unload adds records to the end of the data set. If the DD name represents a member of a PDS and the member exists, it is overwritten.

Instead of providing a data set name with the DD statement, you can dynamically allocate the data set by using one of the following methods. If you dynamically allocate the data set, the value of the +OVERRIDEOUTPUT parameter determines the method that NGT Unload uses.

- Providing the data set name in the XULDDYNM allocation control point.
- Providing the data set name in an OUTPUT command in your unload SYSIN.
  
  You must place the OUTPUT command before the UNLD statement in your SYSIN.

For more information about XULDDYNM, 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*.

**Note**

NGT Unload ignores STMTDDN if you specify FORMAT UNLOAD or FORMAT ASCIIDEL.
**SYSCOPY**

Use this option to instruct NGT Unload to unload from an image copy data set, obtaining the data set name from SYSIBM.SYSCOPY.

If you specify a single partition number, NGT Unload searches only for the indicated partition. Otherwise, NGT Unload performs the search with DSNUM=0.

You can specify one of the following suboptions:

- **FULL**
  
  *(default)* Obtains the full image copy data set that has the highest START_RBA

- **INCR**
  
  Obtains the incremental image copy data set that has the highest START_RBA

- **LAST**
  
  Obtains the image copy data set that has the highest START_RBA

  This data set can be incremental or full.

The following considerations apply to the SYSCOPY option:

- When unloading from an image copy, you cannot specify multiple partitions with the PARTS option.
- If you specify the SYSCOPY option, you cannot specify COPYDDN or COPYDSN.
- Do not use MODE(DB2SQL) or MODE(AUTO) with this option.

**TIMEFMT**

Use this option to override the ULDPARMS parameter for the current unload statement only.

This option enables NGT Unload to generate a variety of time formats other than the standard DB2 formats.

Specify a string of up to 40 characters as an argument for TIMEFMT. NGT Unload replaces character strings to generate the actual time in the formats shown in the following table:
Table 26: Time formats

<table>
<thead>
<tr>
<th>Character string</th>
<th>Time format</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH</td>
<td>Current hour, in either 12-hour or 24-hour format.</td>
</tr>
<tr>
<td>MM</td>
<td>Minutes elapsed from the top of the hour.</td>
</tr>
<tr>
<td>SS</td>
<td>Seconds elapsed from the start of the current minute.</td>
</tr>
<tr>
<td>XM</td>
<td>AM or PM, depending on whether the current time is before or after noon. If XM appears, NGT Unload replaces HH with an hour from a 12-hour clock (00-12); if XM does not appear, NGT Unload replaces HHH with an hour from a 24-hour clock (00-24).</td>
</tr>
</tbody>
</table>

The time element can occur only once, and the entire string must contain at least one time element.

**Example**

For a time of 3:14 in the afternoon, specifying the format string 'HH-MM(XM)' results in 3-14(PM).

**TSFMT**

The TIMESTAMP format overrides the standard DB2 TIMESTAMP format with a variety of new formats.

Specify a string of up to 40 characters as an argument for TSFMT. NGT Unload replaces character strings to generate the actual timestamp in the formats shown in Table 27 on page 152.

Table 27: Timestamp formats

<table>
<thead>
<tr>
<th>Character string</th>
<th>Timestamp format</th>
</tr>
</thead>
<tbody>
<tr>
<td>YYYY</td>
<td>4-digit year.</td>
</tr>
<tr>
<td>YY</td>
<td>2-digit year.</td>
</tr>
<tr>
<td>MM</td>
<td>2-digit month.</td>
</tr>
<tr>
<td>DD</td>
<td>2-digit day of the month.</td>
</tr>
<tr>
<td>HH</td>
<td>Current hour, in either a 12-hour or 24-hour format. See XM below.</td>
</tr>
<tr>
<td>MI</td>
<td>Minutes elapsed from the top of the hour.</td>
</tr>
<tr>
<td>SS</td>
<td>Seconds elapsed from the start of the current minute.</td>
</tr>
<tr>
<td>Character string</td>
<td>Timestamp format</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>XM</td>
<td>AM or PM, depending on whether the current time is before or after noon. If XM appears, NGTUnload replaces HH with an hour from a 12-hour clock (00-12); if XM does not appear, NGTUnload replaces HHH with an hour from a 24-hour clock (00-24).</td>
</tr>
<tr>
<td>NN</td>
<td>2-digit number representing hundredths of a second.</td>
</tr>
<tr>
<td>NNNN</td>
<td>4-digit number representing tenths of a millisecond.</td>
</tr>
<tr>
<td>NNNNNN</td>
<td>6-digit number representing microseconds.</td>
</tr>
</tbody>
</table>

You can set the maximum number of character strings, as follows:

- Maximum of three NN sets
- Maximum of one NNNN and one NN
- Maximum of one NNNNNN

All other characters remain unchanged.

**Note**

There is no corresponding +TSFMT parameter in the ULDPARMS.

---

**UNICODE**

Use this option to tell NGTUnload to use the DB2 installation default (DSNHDECP) CCSIDs to encode the unloaded data in Unicode.

If you do not specify ASCII, UNICODE, or EBCDIC on the unload statement, the default is based on the encoding scheme of the unloaded table, as follows:

- ASCII encodes data using EBCDIC CCSID defined to DB2.
- UNICODE encodes data using EBCDIC CCSID defined to DB2.
- EBCDIC encodes data using EBCDIC CCSID defined to DB2 if you specify MODE(AUTO), but does not encode data if you specify MODE(DIRECT).

**Note**

Do not use MODE(DB2SQL) or MODE(AUTO) with this option.

For more information about page encoding, see “Page encoding (UNLD command)” on page 27.
WHERE

The WHERE clause enables you to specify an SQL-compatible WHERE logical condition.

Note

NGT Unload does not support the VARBINARY and XML data types.

When you use the PARTS option and a WHERE clause in the same UNLOAD statement, the selection of partitions is subject to partition analysis. The resulting unloaded data is the intersection of the results from the specified PARTS option and the results from the WHERE clause.

If the intersection results in no partitions being selected, NGT Unload analyzes the +MINROWS parameter specification. If the +MINROWS criterion is satisfied, the unload continues. If the +MINROWS criterion is not satisfied, NGT Unload issues a warning and the unload job ends with a RC=4.

For more information about the +MINROW parameter, see +MINROWS on page 162.

ZEROPAD

Use this option to pad data in integer, SMALLINT, BIGINT, and decimal columns with zeros for all blank positions.

NGT Unload formats the columns as follows:

- Precedes the data with a "+" or "-" symbol
- Pads to the left

This option applies only to the EXTERNAL format.
Parameters for NGT Unload

This chapter describes the parameters that apply when using the NGT Unload utility.

Overview of NGT Unload parameters

Each NGT utility has a set of parameters that let you control how the utility processes and uses resources. For NGT Unload, you can specify these parameters following the ULDPARMS DD statement in your NGT job stream.

Also available are global parameters that apply to all NGT utilities. Installing an NGT utility configures the defaults for the global parameters, separately for each DB2 subsystem. Each utility job can then override these default parameters if needed. Each utility job's output lists the default parameters and any overrides. For more information about the global parameters, see the *BMC Next Generation Technology General User Guide*.

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.

### +BLANKPAD

Use this parameter to tell NGT Unload to pad VARCHAR columns with blanks instead of low values.

```plaintext
+BLANKPAD — (YES | NO)
```

The BLANKPAD command option overrides this parameter.

**Note**
You cannot use this parameter to not pad VARCHAR columns.

**Operands**

**YES**

Pads VARCHAR columns with blanks instead of low values

**NO**

Pads VARCHAR columns with low values instead of blanks
Example

+BLANKPAD(YES)

NGT Unload pads VARCHAR columns with blanks.

+CHARFLDDLML

Use this parameter to override the default field delimiter characters used for character data fields when unloading using UNLD FORMAT(ASCIIDEL).

The default delimiter for character data is double quotation marks ("). You can use most single characters as a delimiter except a plus sign, comma, or blank. You can also use a null character (that is, no character at all). To do this, code as +CHARFLDDLML(), with no operand.

The UNLD command option FORMAT ASCIIDEL CHARFLDDLML overrides this parameter.

Operands

delim

Single character representing the character field delimiter

You can specify that the delimiter be null.

Example

In the following example, the character strings are not delimited:

+CHARFLDDLML()

In the following example, the character strings are delimited with a period:

+CHARFLDDLML(.)
+CONNECTALL

Use this parameter to tell NGT Unload to connect all table space partitions to the NGT subsystem when unloading selected partitions.

NGT Unload is an online utility and the NGT subsystem (CDBSS) tracks changes during unloads. When NGT Unload runs an unload on a subset of partitions, NGT Unload tracks the changes only in those partitions.

It is most efficient to track only changes for the subset of partitions that you are unloading using +CONNECTALL(NO). However, this prevents NGT Unload from unloading separate or overlapping partitions concurrently in a separate unload job.

Use +CONNECTALL(Y) when you expect to run multiple unload jobs on the same partitioned table simultaneously and you are not unloading all partitions.

Operands

YES or Y

Connects all partitions even if the WHERE clause limits the partitions being unloaded

NO or N

Connects only the partitions selected for unloading in the WHERE clause

This is the default option.

Example

In the following example, track changes for all partitions in preparation for concurrent unloads on the table:

+CONNECTALL(Y)

In the following example, track changes only for the specific partitions that you are unloading:

+CONNECTALL(N)
Use this parameter to set or override the default date format.

The distributed default is ISO. This default affects only the external representation of date values when using any format other than UNLOAD.

**Operands**

ISO

Uses the ISO date format (yyyy-mm-dd)

EUR

Uses the European date format (dd/mm/yyyy)

JIS

Uses the Japanese Industrial Standard date format (yyy-mm-dd)

USA

Uses the USA date format (mm/dd/yyyy)

DECP

Uses the standard date format as defined in DSNHDECP

**Example**

In the following example, NGT Unload sets the default format to European:

```
+DATEFMT(EUR)
```
Use this parameter to specify the number of successive unload statements for the same table that NGT Unload can process concurrently. It performs this with one read of the table and parallel writing of the various unloads to separate unload data sets.

NGT Unload processes unload statements in parallel if the following conditions in those statements are satisfied:

- The value of `+DEGREE` is greater than 1.
- You do not specify `ORDER BY`.
- You do not specify partition numbers.
- You do not specify `BYPART`.
- You are not unloading from an image copy.
- You do not specify `UNLOAD DIRECT NO` or `UNLD MODE(DB2SQL)`.
- Each unload statement unloads to a different unload data set.
- There is no mix of `SHRLEVEL CHANGE` (dirty unload) and `CONCURRENT` (online unload).
- All unloads are to either statically allocated unload data sets or dynamically allocated data sets, but not to a mix of both.
- The size of all merged unload statements to be processed in parallel does not exceed 63 KB. If the size exceeds 63 KB, NGT Unload processes the first 63 KB in parallel and then the remainder in parallel.
- One of the following specifications:
  
  - Use disk or VTS unload files (recommended). Specify `MEDIA=D` in the `XULDDYNM` automation control point, and copy to disk or VTS.
  
  - If you must unload to tape, specify `FSEQ=1` in `XULDDYNM` to prevent stacking and specify `DDNCOUNT(n)` in `NGTTAPE` to allocate `n` tape drives to match `+DEGREE(n)`. Attempts to stack data sets on VTS or tape result in an error.

Successive statements that satisfy these conditions form a parallel output group. If a statement does not satisfy these conditions, the existing group ends and another group starts. NGT Unload dispatches multiple groups simultaneously depending on `+MAXTASKS`.
The NGT Unload reader can read a maximum of 32 partitions concurrently and process up to 63 output data sets in parallel.

Grouping has a number of advantages:

- Reduces CPU elapsed and connect time
- Reduces memory utilization
- Reduces the number of EXCP counts

However, you cannot stack two output data sets in a single group on the same tape.

When you combine two statements, NGT Unload issues a message such as the following message (during parse):

```
NGTN069 UNLD STMT 1 has been merged with STMT 2. Queue Depth=2
```

NGT Unload issues a message such as the following message before starting work:

```
NGTN070 Dispatching 10 concurrent reader(s) using 22 buffers to 5 parallel writer(s)
```

**Operands**

$n$

Between 0 and 63

**Example**

The following example processes up to four successive unload statements in parallel with one read of the table:

```
+DEGREE(4)
```

```
+FIELDESEP
```

Use this parameter to override the default field separator character (a comma) between fields when unloading using UNLD FORMAT(ASCIIDEL).

You can use any single character as a separator character except for a plus sign. You can use a blank as a separator.
The UNLD command option FORMAT ASCIIDEL FIELDSEP overrides this parameter.

**Operands**

*char*

A single character representing a field separator

**Example**

In the following example, NGT Unload separates all fields with a hyphen:

```
+FIELDSEP(-)
```

In both of the following examples, NGT Unload places a blank between all fields:

```
+FIELDSEP()
+FIELDSEP( )
```

**+MINROWS**

If you use this parameter and the number of rows unloaded is less than MINROWS, NGT Unload issues a message that includes the number of rows unloaded and the MINROW values. In this case, the unload ends with `RC=4`.

```
+MINROWS (0 integer)
```

**Operands**

*0*

Processing retrieves all rows

This is the default setting.

*integer*

An integer that represents the minimum number of rows to retrieve
Example

In the following example, processing unloads at least 200 rows or the job ends with a warning:

```
+MINROWS(200)
```

**+NUMFLDDLM**

Use this parameter to override the default field delimiter character used for numeric data fields when unloading using UNLD FORMAT(ASCIIDEL).

The default action for placing a delimiter around numeric fields is not to delimit numeric fields. You can use most single characters as a delimiter except for the plus sign (+), a comma (,), or a blank. You do not need to use any character as a delimiter.

**Operands**

`null`

The default value

Indicate a null value by specifying nothing between the parentheses:

```
+NUMFLDDLM()
```

`delimiter`

A single character that represents the numeric field delimiter

**Examples**

In the following example, NGT Unload does not delimit numeric strings:

```
+NUMFLDDLM()
```

In the following example, NGT Unload delimits numeric strings with the USA dollar currency sign.

```
+NUMFLDDLM($)```
Use this parameter to set or override the default time format.

The distributed default is ISO. This default affects only the external representation of date values when using any format except UNLOAD.

**Operands**

- **ISO**
  
  Uses the ISO time format

- **EUR**
  
  Uses the European time format

- **JIS**
  
  Uses the Japanese Industrial Standard time format

- **USA**
  
  Uses the USA time format

- **DECP**
  
  Uses the standard time format as defined in DSNHDECP

**Example**

In the following example, NGT Unload sets the default format to ISO:

```
+TIMEFMT(ISO)
```
Statement examples for NGT Unload

This chapter provides examples of commands that control how the NGT Unload utility unloads database objects to sequential data sets.

Example: unloading by using a table lookup

It is sometimes useful to select the rows to unload by determining if the rows exist in another table (for example, if you have a production application that identifies rows to be unloaded over the weekend for later transfer to a non-DB2 application).

For an example, see the following unload statement:

```
//SYSIN DD *
UNLOAD DATA FORMAT(DSNTIAUL) UNLOADDN(SYSREC)
SELECT *
FROM TABLE BMC.PRODUCTION_TBL
WHERE RECID IN (SELECT RECID FROM ROWS2UNLD)
```

Example: unloading a specific number of rows

In the following example, NGT Unload stops unloading at 25,000 rows:

```
//SYSIN DD *
UNLOAD FORMAT(UNLOAD) UNLOADDN(SYSREC)
SELECT *
FROM TABLE DBX14.RFID_CODES
LIMIT 25000
```

**Note**

For more information about the LIMIT option, see “LIMIT” on page 72.
Example: unloading from a copy

You can use the INFILE option to unload from a full image copy data set that is registered in the DB2 catalog.

Specifying INFILE IMAGECOPY unloads the most recent full image copy data set that NGT Unload finds in the SYSIBM.SYSCOPY table or, for a cabinet copy, the BMCXCOPY table. The following example also illustrates the following functions:

- Overriding the default DB2 date and timestamp format
- Unloading from a subset of a copy
- Sorting the data from the copy

```
//SYSIN DD *
UNLOAD INFILE IMAGECOPY
  TSFMT('MM.DD.YY/HH:MM:SS.NN')
  DATEFMT('MM.DD.YY')
  PART 3
  SELECT *
    FROM BMC.EMPLS WHERE SSN > '200000000'
    ORDER BY HIRE_DATE DESC, EMPL_LNAME
```

Example: unloading a LOB table space

This example names an unload file to contain both the unloaded base table space data and the associated LOB table space data. Using one unload file prevents some parallelism and the use of servers, which spreads the workload and increases processing speed.

**Note**

Beginning with NGT Unload version 12.1, you do not need a license for NGT LOBMaster to unload LOB data.

```
//SYSREC DD UNIT=SYSDA,DISP=(NEW,CATLG,DELETE),
  DSN=HLQ.UNLD.LOBTABLE,
  SPACE=(CYL,(1000,500),RLSE)
//SYSIN DD *
UNLOAD FORMAT DSNTIAUL UNLOADDN SYSREC
  SPANNED YES
  SELECT *
    FROM TABLE OWNRID.Table_With_LOB
```
Output file formats

This chapter describes NGT Unload output file formats.

LOAD statements

You can automatically generate LOAD statements to a defined SYSCNTL or SYSPUNCH DD card when you specify certain values for the FORMAT option. You can then use the resulting statement to reload data in another job.

The following table lists which formats generate LOAD statements in a SYSCNTL or SYSPUNCH DD:

Table 28: Generating LOAD statements in SYSCNTL or SYSPUNCH DD

<table>
<thead>
<tr>
<th>FORMAT keyword</th>
<th>Generates a LOAD statement in a SYSCNTL or SYSPUNCH DD?</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCIIDEL</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>This value is valid only with UNLD.</td>
</tr>
<tr>
<td>DELIMITED</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>This value is valid only with UNLD.</td>
</tr>
<tr>
<td>DSNTIAUL</td>
<td>Yes</td>
</tr>
<tr>
<td>EXTERNAL</td>
<td>Yes</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>No</td>
</tr>
</tbody>
</table>

To generate a LOAD statement and unload the data from the table for a particular output format, allocate the SYSCNTL DD or SYSPUNCH DD to a sequential file. For example:

//SYSCNTL DD SYSOUT=*
or

//SYSCNTL DD DISP=SHR,DSN=NGT.UNLOAD.AUTO.GENERATE

To override the SYSCNTL DD name, use the CNTLDDN option on the UNLOAD command. To override the SYSPUNCH DD name, use the STMTDDN option on the UNLD command.

Instead of providing a data set name with the DD statement, you can dynamically allocate the data set by using one of the following methods. If you dynamically allocate the data set, the value of the +OVERRIDEOUTPUT parameter determines the method that NGT Unload uses.

- Providing the data set name in the XULDDYNM automation control point.
- Providing the data set name in an OUTPUT command in your unload SYSIN.

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

For more information about XULDDYNM, 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*.

If SYSCNTL, SYSPUNCH, or an alternative DD is not allocated, NGT Unload assumes that you want to bypass generation of the LOAD statement and does not issue an error message.

---

**NGT Unload output file formats**

This topic describes the various output data formats supported by NGT Unload.

You can specify an output format using one of the FORMAT keyword operands shown in the following table:

**Table 29: Operands and output formats**

<table>
<thead>
<tr>
<th>Operand</th>
<th>Available with UNLOAD command</th>
<th>Available with UNLD command</th>
<th>Output format</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSNTIAUL</td>
<td>X</td>
<td>X</td>
<td>“DSNTIAUL format” on page 169</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>X</td>
<td>X</td>
<td>“UNLOAD format” on page 171</td>
</tr>
<tr>
<td>DELIMITED</td>
<td>X</td>
<td></td>
<td>“DELIMITED format” on page 173</td>
</tr>
<tr>
<td>ASCIIDEL</td>
<td>X</td>
<td></td>
<td>“ASCIIDEL format” on page 174</td>
</tr>
</tbody>
</table>
With all output formats, you can unload all the columns or a subset of the columns.

**Examples**

In the examples that appear in the following topics, make the following assumptions:

- The table is defined by the following DDL:

  ```sql
  CREATE TABLE NGT.TABLE1
  (  NAME      CHAR(6)  NOT NULL ,
     AGE       INTEGER  NOT NULL ,
     SALARY    DECIMAL (6,2 ) ,
     COMMENT   VARCHAR (20) ) IN DATABASE NGT ;
  ```

- The three rows of data is represented by the following DDL:

  ```sql
  INSERT INTO NGT.TABLE1 VALUES ( 'TODD' ,  16  ,  123.45 ,  'USE UNLOAD!!' ) ;
  INSERT INTO NGT.TABLE1 VALUES ( 'MATTEO' ,  32  ,  9500.50 ,  NULL ) ;
  INSERT INTO NGT.TABLE1 VALUES ( 'IRINA' ,  48  ,  NULL ,  ''  ) ;
  ```

**DSNTIAUL format**

The NGT Unload DSNTIAUL format is the same as the IBM DSNTIAUL format.

By default, NGT Unload pads variable length fields to their maximum length with binary zeros. You can override this. NGT Unload converts DATE, TIME, and TIMESTAMP fields to character strings.

If a column is nullable, a NULL marker immediately precedes or follows the column value in the output file. This NULL marker may consist of a single character or of multiple characters. A maximum of four characters is allowed.

If the NULL marker is binary zeros (X'00'), the column value was not null when unloaded. If the NULL marker is not binary zeros, the column value was null when unloaded and the area in the output row represented by that column contains binary zeros.

NGT Unload converts numeric values from internal DB2 format to S/390 data type formats.

Each variable-length column is prefixed by a two-byte number that represents the column length excluding both the prefix itself and the null indicator field (if present).
Depending on the NULL field specifications, the null indicator byte either immediately precedes or follows the nullable field.

You can set the RECFM of the output file when using the DSNTIAUL format using one of the following methods:

- If you are using the UNLD command, specify the PAD or NOPAD option.
  
  If you specify NOPAD, the RECFM of the output file is set to VB. This keyword does not pad the area to the right of the data value in a varying field to binary zeros.
  
  You can also set the record format to VB by specifying `DCB=RECFM=VB` in the DD statement for the unload data set. NGT Unload does not pad the unloaded variable fields with binary zeros. This is incompatible with specifying the PAD keyword.

- Override the RECFM value in the JCL for the unload data set.
  
  If you do not override the RECFM parameter in the JCL, specifying the PAD keyword sets the RECFM to FB. This keyword sets the unused area to the right of the data value in a varying field to binary zeros.

**Note**

The DSNTIAUL keyword tells NGT Unload to automatically generate a LOAD statement to a defined SYSCNTL or SYSPUNCH DD card. For more information, see “LOAD statements” on page 167.

### Example: using the DSNTIAUL format

To have NGT Unload unload table NGT.TABLE1 using the DSNTIAUL format to a file with record format FB, use the following statement:

```
UNLOAD FORMAT DSNTIAUL FROM TABLE NGT.TABLE1
```

The results appear as follows:

<table>
<thead>
<tr>
<th>1</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
<th>35</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLS/NULLS:</td>
<td></td>
<td></td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| ROW 1:      TODD  ...........USE UNLOAD!!.........
  (EDCC4400010135000EEC4EDDDC55000000000)
  3644000000024C00CE250453614AA000000000
| ROW 2:      MATTEO...............................?
  (DCEECD00020500000000000000000000000006)
  4133560000905C00000000000000000000000C
| ROW 3:      IRINA ........?.......................
  (CDCDC4000300006000000000000000000000)
  99951000000000C00000000000000000000000

For information about the assumptions in this example, see “NGT Unload output file formats” on page 168.
In these results:

- A vertical bar (|) represents the start of each column in the row labeled COLS/NULLS
- An asterisk (*) represents null indicators.

Descriptions of the fields and their values are shown in the following table:

**Table 30: Fields and values**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>TODD, 6 bytes</td>
</tr>
<tr>
<td>AGE</td>
<td>X'000000010', 4 bytes</td>
</tr>
<tr>
<td>SALARY</td>
<td>X'0012345C', 4 packed decimal bytes</td>
</tr>
<tr>
<td></td>
<td>X'00', a 1-byte null flag indicating NOT NULL</td>
</tr>
<tr>
<td>COMMENT</td>
<td>X'000C', a 2-byte length field</td>
</tr>
<tr>
<td></td>
<td>USE UNLOAD!!, 12 bytes of characters</td>
</tr>
<tr>
<td></td>
<td>X'0000000000000000', 8-bytes binary zero fill bytes</td>
</tr>
<tr>
<td></td>
<td>X'00', a 1-byte null flag</td>
</tr>
<tr>
<td>ROW 2, COMMENT column</td>
<td>This column varies in length, always at offset 15 in the row. The length in ROW 2 shows a zero length. The null indicator, always at offset 37 in the row, shows a ?, indicating that this field is null.</td>
</tr>
<tr>
<td>ROW 3, SALARY column</td>
<td>This field is null (as for ROW 2, COMMENT column). The ROW 3, SALARY column is null. SALARY is at offsets 10 through 13. Offset 14 is the null indicator and shows as null.</td>
</tr>
<tr>
<td>ROW 3, COMMENT column</td>
<td>The entire field is binary zeros, including the length and the null indicator. This means that the value is NOT NULL, and has a varying string length of zero length.</td>
</tr>
</tbody>
</table>

**UNLOAD format**

The NGT Unload UNLOAD format is the same as the format that the IBM DB2 REORG utility uses with the UNLOAD PAUSE option.

This format operates based on the following rules:

- RECFM is always VB.
- The expansion of varying fields to their full length is based on the use of the PAD and NOPAD keywords (if you are using the UNLD command).
- Variable-length field length bytes indicate the actual length of the data, not the maximum allowed length.

- Variable-length fields are prefixed with a 2-byte field containing the length of the field (excluding the prefix itself).

- Column values stay in DB2 internal format.

- Each row contains a 6-byte record prefix that contains a 1-byte row flag, a 2-byte length (the same for every row), a 2-byte OBID for the owning table, and a 1-byte MAP ID reference.

- Null indicators precede nullable columns. NGT Unload copies them without changes from the values in the table being unloaded.

- You can override the blocksize selected by NGT Unload by coding the blocksize keyword in the JCL.

**Example: using the UNLOAD format**

The following unload statement:

```
UNLOAD FORMAT(UNLOAD) FROM TABLE NGT.TABLE1
```

produces the following output file:

<table>
<thead>
<tr>
<th>1</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
<th>35</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>COLS/NULLS:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROW 1:</td>
<td>TODD</td>
<td>USE UNLOAD!!</td>
<td>(002000EDCC44000100135000EEC4EDDDCC5500000000)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>00B03136440000000024C0D0E250453614AA0000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROW 2:</td>
<td>MATTEO</td>
<td></td>
<td>(002000DCEEC0020050000F00000000000000000)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>00B0324133560000905C01F000000000000000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROW 3:</td>
<td>IRINA</td>
<td></td>
<td>(002000CDCDC40003F000000000000000000000000)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>00B033999510000F000001000000000000000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For information about the assumptions in this example, see “NGT Unload output file formats” on page 168.

In these results:

- A vertical bar (|) represents the start of each column in the row labeled COLS/NULLS.
- An asterisk (*) represents null indicators.

Descriptions of the fields and their values are shown in the following table:
Table 31: Columns and values

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field 1, Record ID</td>
<td>X'00', 1 byte.</td>
</tr>
<tr>
<td>Field 2, Record Length</td>
<td>X'002B', 2 bytes long. This is the total length of the record.</td>
</tr>
<tr>
<td>Field 3, Record OBID</td>
<td>X'0003', 2 bytes.</td>
</tr>
<tr>
<td>Field 4, MAP ID Reference</td>
<td>X'01', 1 byte.</td>
</tr>
<tr>
<td>NAME</td>
<td>TODD, 6 bytes.</td>
</tr>
<tr>
<td>AGE</td>
<td>X'00000010', 4 bytes.</td>
</tr>
<tr>
<td>SALARY</td>
<td>X'00', the 1-byte null indicator, followed by X'0012345C', is 4 packed decimal bytes. The field is NOT NULL.</td>
</tr>
</tbody>
</table>
| COMMENT            | ■ X'000D', a 2-byte length field, that includes the length of the null indicator  
                     ■ X'00', a 1-byte null flag  
                     ■ USE UNLOAD!!, 12 bytes of characters  
                     ■ X'0000000000000000', 8-bytes binary zero fill bytes |
| ROW 2, COMMENT column| Always at offset 21 in the row, this column varies in length. The length in ROW 2 shows a length of 1 byte. Of this length, the null indicator is 1 byte and the rest is data. This field is null, so there is no data. |
| ROW 3, SALARY column| This field is null (as for ROW 2, COMMENT column).                     |
| ROW 3, COMMENT column| The field is 1 byte in length, which includes only the null indicator. In this example, there is a varying length string of zero length. |

**DELIMITED format**

In this format, unload data files are in a delimited format.

*Note* The UNLOAD command does not currently have an option to unload data files 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 hexadecimal constant. You can specify either COLDEL ‘#’ or COLDEL X’23’. For Unicode and ASCII encodings, you should specify the delimiter character in the utility statement as a hexadecimal constant, or the result can be unpredictable.

You can use the following delimiters with FORMAT(DELIMITED) on the UNLD command. For more information, see “FORMAT” on page 135.

- COLDEL specifies the column delimiter that is used in the unload file.
- CHARDEL specifies the character string delimiter that is used in the unload file.
- DECPT specifies the decimal point character that is used in the unload file.

You cannot specify the same character for more than one type of delimiter.

**ASCIIDEL format**

The NGT Unload ASCIIDEL format unloads all fields to character format. This keyword is available only with the UNLD command.

*Note*

The ASCIIDEL keyword does not automatically generate a LOAD statement in a SYSPUNCH DD.

By default, NGT Unload delimits character fields within double quotation marks. It does not delimit numeric fields and separates each field with a comma.

**Example: using the ASCIIDEL format**

The following unload statement:

```sql
UNLD DATA FORMAT(ASCIIDEL) FROM TABLE NGT.TABLE1
```

produces the following output file:

```
1      5     10    15    20    25    30    35    40
+------------------------+------------------------+
ROW 1: "TODD",16,123.45, "USE UNLOAD!!"
ROW 2: "MATTEO",32,9500.00,"
ROW 3: "IRINA",48,""
```

For information about the assumptions in this example, see “NGT Unload output file formats” on page 168.

By default, null character data is represented by an empty string, or enclosed by two double quotation marks. Null numeric fields receive no data, as is shown in ROW 3.
Use NGT Unload parameters to change delimiters, as shown in the following table:

### Table 32: Using parameters to change delimiters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>+NUMFLDDLM</td>
<td>To change the field delimiters for numeric data</td>
</tr>
<tr>
<td>+CHARFLDDLM</td>
<td>To change the field delimiters for character data</td>
</tr>
<tr>
<td>+FIELDSEP</td>
<td>To change the character used to separate fields from the default of comma (,) to any other character</td>
</tr>
</tbody>
</table>

These parameters affect all the unload statements in the job. You can also change them for a specific unload statement by using the UNLD keywords FIELDSEP, CHARFLDDLM, or NULLSTRING. For more information, see “ASCIIDEL” on page 137.

### EXTERNAL format

This format is similar to the DSNTIAUL format in DB2, except that NGT Unload unloads numeric fields in right-justified external format.

**Note**

In this format, a SYSPUNCH or SYSCNTL file (if allocated) contains a LOAD statement for the generated file.

#### Note

The NULL marker immediately precedes or follows nullable columns. This marker may consist of a single character or a string of up to four characters.

A NULL marker of binary zeros (X'00') indicates that the column value was not null when unloaded.

A NULL marker without binary zeros indicates that the column value was null when unloaded. The area in the output row represented by that column contains binary zeros.

Table 33 on page 175 shows the maximum length for different numeric data types.

### Table 33: Data types and maximum lengths

<table>
<thead>
<tr>
<th>Data type</th>
<th>Maximum length</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMALLINT</td>
<td>6 bytes</td>
</tr>
<tr>
<td>INTEGER</td>
<td>11 bytes</td>
</tr>
<tr>
<td>DECIMAL (p,s)</td>
<td>The precision plus 2 bytes</td>
</tr>
<tr>
<td>Data type</td>
<td>Maximum length</td>
</tr>
<tr>
<td>------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>FLOAT(21)</td>
<td>15 bytes</td>
</tr>
<tr>
<td>FLOAT(23)</td>
<td>23 bytes</td>
</tr>
<tr>
<td>BIGINT</td>
<td>20 bytes</td>
</tr>
<tr>
<td>DECFLOAT(16)</td>
<td>23 bytes</td>
</tr>
<tr>
<td>DECFLOAT(34)</td>
<td>42 bytes</td>
</tr>
</tbody>
</table>
Recommended automation control points for NGT Unload

NGT automation control points, which are integrated into all NGT utilities, can enhance and expand processing options. This chapter describes recommended automation control points for NGT Unload.

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 Unload 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 Unload. However, you can also use other automation control points. For more information about automation control points, see the *NGT Automation Reference Manual*.

**XSUTGLOB**

Use to set global variables that are then used by other automation control points

**XSUT0000**

Use to abort a run before it starts

This automation control point is called before any processing starts.
**XSUTSYIN**

Use to modify SYSIN, enabling you to override or disallow some input parameters.

This automation control point is called before the parser reads SYSIN and has full access to the SYSIN.

**XSUTTERM**

Use to insert run statistics into a DB2 table or to automatically send an audit report as an e-mail at the completion of processing.

This automation control point is called after processing has finished.

**XSVRXERR**

Use to alert a user or to raise an error flag.

This automation control point is called if processing ends with an error condition.
ROWEXIT module

The ROWEXIT module is an assembler language API that enables you to call a specified user exit from the NGT Unload utility each time the utility unloads a row from the DB2 table.

NGT Unload contains an 8-byte token area defined in the parameter structure for each call. The token is reserved for use by the user exit. The token value passes between the ROWEXIT module and the user exit. After this value is set, the ROWEXIT module cannot change it. Therefore, the user exit has a modifiable storage area which remains unchanged across ROWEXIT module calls.

Note

The user token field and the pointer field (row description block or row pointer) are 8-byte fields. This ensures that they are suitable for 64-bit architecture and 8-byte pointers. If a 31-bit environment is in use, use only the last 4 bytes of the CDBROWPTR field for addressability to the table row header or row descriptor block information, or to the row pointer.

For information about the ROWEXIT keyword, see ROWEXIT on page 148.

The ROWEXIT module sends the following types of calls to the user exit:

- Initialization call on page 180
- Processing call on page 180
- Termination call on page 181

Each call has a specific set of parameters and returns specific values.

The user exit performs the requested process for each call, including setting the row disposition value and return code value. The user exit then returns these values to the ROWEXIT module, which examines and processes them.
Initialization call

This type of call establishes a dialog session with the user exit and identifies to the user exit the characteristics of the row being processed.

Processing call

This type of call passes the row to the user exit.

- **Row disposition value**
  The user exit examines the row disposition value first. This value indicates the type of processing request for the current row.

- **Return code value**
  This value indicates the success or failure of the call. If this value is 8 or greater, processing terminates after the handling of the current row.

The user exit examines the row and returns to the ROWEXIT module one of the following requests for that row:

- Process the returned *unmodified row* in the same way that NGT Unload normally processes rows.
- Process the returned *modified row* in the same way that NGT Unload normally processes rows.
- Exclude the row from normal NGT Unload processing.

The user exit can alter the contents of rows, subject to the following limitations:

- It cannot change the length of a fixed length row.
- It can change the length of a varying length row, but only the varying length fields or columns within that row. The length cannot exceed the maximum field or column length.
- The user exit should use caution when altering field or column contents, as subsequent processing might have data type expectations for the field or column value.

---

**Note**

Processing calls belong to a single table in the current table being processed.
Termination call

This type of call ends a dialog session between the ROWEXIT module and the user exit.

ESTAE routine

This topic provides information about the ESTAE routine applicable to the ROWEXIT module and user exit.

The user exit is called authorized, in user key 8. NGT Unload creates an ESTAE routine before calling the exit for recovery purposes. If an abend occurs within the user exit, processing terminates and the user exit does not examine the row disposition value. The user exit can establish its own recovery environment, but should cancel all recovery and retry routines before passing control back to the exit processor.

NGT Unload establishes the ESTAE routine unconditionally for an exit during an initialization call. After making the initialization call, you can turn off the ESTAE routine through a field in the parameter structure. If you disable the ESTAE capability, the table processing overhead decreases, but the ROWEXIT module cannot intercept errors if the user exit abends.

You should prepare the user exit to accept the parameter list and fields in storage that contain 31-bit addresses. The user exit has no resource consumption restrictions, such as CPU limits, and may connect to the DB2 SSID passed in the table row header block. The user exit is responsible for properly maintaining and terminating its connection to DB2.

Exit register conventions at entry

At entry to processing, the user exit sets the registers as follows:

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0</td>
<td>Unpredictable contents</td>
</tr>
<tr>
<td>R1</td>
<td>Pointer to a parameter list block</td>
</tr>
<tr>
<td>R2–R12</td>
<td>Unpredictable contents</td>
</tr>
<tr>
<td>R13</td>
<td>Pointer to save area</td>
</tr>
<tr>
<td>R14</td>
<td>Return address</td>
</tr>
</tbody>
</table>
Exit register conventions upon return

Upon return from processing, the user exit sets the registers as follows:

<table>
<thead>
<tr>
<th>Register</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0</td>
<td>Unpredictable contents</td>
</tr>
<tr>
<td>R1</td>
<td>Pointer to a parameter list block</td>
</tr>
<tr>
<td>R2–R12</td>
<td>Unpredictable contents</td>
</tr>
<tr>
<td>R13</td>
<td>Register save area</td>
</tr>
<tr>
<td>R14</td>
<td>Unpredictable contents</td>
</tr>
<tr>
<td>R15</td>
<td>Return code (also found in CDBRETCODE in the parameter list block)</td>
</tr>
</tbody>
</table>

Initialization call specifications

This topic contains initialization call input parameters and output values.

Input parameters

Register 1 points to the following initialization parameters (in the format +offset(length)):

+0(4)

Function code of 0 which identifies the initialization call

+4(4)

Row disposition which is initialized to a value of -1.

+8(4)

Return code value which is initialized to a value of -1
+12(4)

ESTAE request value which is initialized to a value of 1

+16(8)

User token area that the user exit sets

+24(8)

Pointer which points to the structure identifying the row format as defined by the table being processed

This pointer refers to data that the table row header maps, followed by the row descriptor block structure. This data structure identifies one table row header structure only, followed by multiple row descriptor block structures. The number of row descriptor block structures corresponds to the number of columns in the table that NGT Unload is processing.

+32(82)

Message text area which is an 82-byte area that holds messages created by user exit

+114(2)

Reserved for future use

+116(4)

Reserved for future use

The high-order byte of this value is set to X'80'.

**Return values**

Register 1 points to the following values upon return from the initialization call:

+0(4)

Function code which the user exit does not examine upon return

+4(4)

Row disposition value which the user exit does not examine upon return

+8(4)

Return code value
0

Indicates a successful initialization call

4

Indicates that unload processing should continue normally, without any further calls to the user exit

Only the termination call is processed after this.

>4

Indicates an unsuccessful initialization call. Processing terminates for non-zero values

If the message text area contains a message, the ROWEXIT module issues it upon return from the user exit.

+12(4)

ESTAE request value

0

Indicates that no ESTAE routine is created for subsequent ROWEXIT calls

1

Indicates that the ESTAE routine will continue to be created for subsequent ROWEXIT calls

+16(8)

User token area that the user exit sets

+24(8)

Pointer to the table row header or row descriptor block structure that the user exit does not examine upon return

+32(82)

Message text area which is an 82-byte area that holds messages created by the user exit

The first 2 bytes of this field represent the message length and does not include the 2-byte length field. NGT Unload automatically prints this message if the return code value is 8 or higher.
The high-order byte of this value is set to X'80'.

**Processing call specifications**

This topic contains processing call input parameters and output values.

**Input parameters**

Register 1 points to the following processing parameters:

+0(4)

Function code of 4 which identifies the processing call

+4(4)

Row disposition which is initialized to a value of -1

+8(4)

Return code value which is initialized to a value of -1

+12(4)

ESTAE request value which is the same as the value for the previous ROWEXIT call

+16(8)

User token area that the user exit sets and the ROWEXIT module does not modify

+24(8)

Pointer to the current row being processed
+32(82)
Message text area which is an 82-byte area that holds messages created by the user exit

+114(2)
Reserved for future use

+116(4)
Reserved for future use.
The high-order byte of this value is set to X'80'.

**Return values**

Register 1 points to the following values upon return from the process call:

+0(4)
Function code of 4 which remains unchanged

+4(4)
Row disposition value

0
Indicates that NGT Unload processing continues as normal

NGT Unload processes the unmodified row as returned from the user exit.

4
Indicates that NGT Unload processing continues as normal, even though the user exit has modified the row

The row passed to the user exit is the row in the output buffer. Upon return from the user exit, NGT Unload does not validate the row value. If the modifications that the user exit makes to row cause an overlay, the results are unpredictable. Therefore, be very careful when you modify the row.

8
Indicates that this row will be skipped and excluded from normal NGT Unload processing
Return code value

0

Indicates a successful processing call

4

Indicates that ROWEXIT module processing should continue normally, without any further calls to the user exit.

Only the termination call is processed after this.

8

Indicates an unsuccessful processing call

Processing terminates for non-zero values. If the message text area contains a message, the ROWEXIT module issues it upon return from the user exit.

ESTAE request value

0

Indicates that no ESTAE routine is created for subsequent ROWEXIT calls

1

Indicates that the ESTAE routine will continue to be created for subsequent ROWEXIT calls

User token area that the user exit sets and ROWEXIT module has not modified

The address of the row that is currently being processed

If you intend to modify this row, set this value to the address of the modified row.

The first 2 bytes of this modified row must contain a length value that correctly reflects the modifications made to that row. The modified row
length must not exceed the maximum row length, as identified in the row descriptor block referenced in the initialization call.

+32(82)

Message text area which is an 82-byte area that holds messages created by user exit

The first 2 bytes of this field represent the message length and does not include the 2-byte LL field. This message is automatically printed if the return code value is 8 or higher.

+114(2)

Reserved for future use

+116(4)

Reserved for future use

The high-order byte of this value is set to X'80'.

Termination call specifications

This topic contains termination input parameters and output values.

Input parameters

Register 1 points to the following termination parameters:

+0(4)

Function code of 8 which identifies the termination call

+4(4)

Row disposition which is a full word of zeros

+8(4)

Return code value which is a full word of zeros

+12(4)

ESTAE request value which is the same as the value for the previous ROWEXIT call
User token area that the user exit sets and the ROWEXIT module does not modify

Pointer to the current row being processed which is a full word of zeros

Message text area which is an 82-byte area that holds messages created by the user exit

Reserved for future use

Reserved for future use. The high-order byte of this value is set to X'80'

**Return values**

Register 1 points to the following values upon return from the process call:

Function code of 8 which remains unchanged

Row disposition value which the user exit does not examine for termination calls

Return code value

0

Indicates a successful termination call

>0

Indicates an unsuccessful termination call

NGT Unload processing terminates immediately for all non-zero values.
If the message text area contains a message, the ROWEXIT module prints it upon return from the user exit.

8

Indicates that this row will be skipped and excluded from normal NGTUnload processing

+12(4)

ESTAE request value

+16(8)

User token area that the user exit sets and ROWEXIT module has not modified

+24(8)

The address of the row that is currently being processed

The user exit does not examine this value.

+32(82)

Message text area which is an 82-byte area that holds messages created by user exit

The first 2 bytes of this field represent the message length and does not include the 2-byte LL field. This message is automatically printed if the return code value is 8 or higher.

+114(2)

Reserved for future use

+116(4)

Reserved for future use

The high-order byte of this value is set to X’80’.

**Detailed design**

The following topics describe the control blocks and data structures that pass from the ROWEXIT module to the user exit.
Parameter structure

This topic describes the parameter structure contained in register 1 for all ROWEXIT module initialization, processing, and termination calls.

+0 FUNCTION

This full word identifies the function request.

0

Identifies the initialization call

4

Identifies the processing call

8

Identifies the termination call

+4 ROW DISPOSITION

Identifies the function to be performed for the current row being processed

The user exit examines this value only for processing calls and ignores the value for the initialization and termination calls.

0

Informs the ROWEXIT module to process the unmodified row as normal

4

Informs the ROWEXIT module that the row has been modified and specifies that the ROWEXIT module should process the updated row as normal

8

Informs the ROWEXIT module to skip this row to prevent normal processing

+8 RETURN CODE

This full word identifies the return code value that the user exit sets.

0

Specifies that processing should continue after the current row is processed
4

Specifies that processing should continue as normal, without any further calls to the user exit

8

Specifies that processing should terminate after the current row is processed

+12 ESTAE REQUEST

This full word identifies the request from the user exit to continue or to discontinue the creation of the ESTAE routine for subsequent user exit calls.

0

Specifies that no ESTAE route should be created for subsequent user exit calls

1

Specifies that an ESTAE routine should be created for subsequent user exit calls

This is the default value for this field before the initialization call.

+16 USER TOKEN

The user exit can use this double word for its processing. The user exit can populate this value; the ROWEXIT module does not modify this value.

+24 ROW VALUE

For initialization calls, this double word points to the table row header or row descriptor block information.

For processing calls, this value points to the row value itself. If you want the user exit to modify the row value, ensure that this value points to the modified row value upon return from the processing call of the user exit.

+32 MESSAGE TEXT

This 82-byte field holds the message that the user exit creates. The user exits prints this only if the return code value is 8 or higher. The first 2 bytes of this field represent the message length and do not include the 2-byte length field.

+114 RESERVED1

Unused 2 bytes of storage
Assembler DSECT

The following DSECT represents the ROWEXIT module parameter structure:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDBFCODE</td>
<td>DS 0D.A</td>
<td>Function Code</td>
</tr>
<tr>
<td>CDBROWDISP</td>
<td>DS A</td>
<td>Row Disposition Code</td>
</tr>
<tr>
<td>CDBRETCODE</td>
<td>DS A</td>
<td>Exit Return Code</td>
</tr>
<tr>
<td>CDBESTAE</td>
<td>DS A</td>
<td>ESTAE Request Value</td>
</tr>
<tr>
<td>CDBUSERTOK</td>
<td>DS D</td>
<td>Optional User Token</td>
</tr>
<tr>
<td>CDBROWPTR</td>
<td>DS D</td>
<td>Pointer to Row Structure or Row</td>
</tr>
<tr>
<td>CDBMESSAGE</td>
<td>DS XL82</td>
<td>User exit message</td>
</tr>
<tr>
<td>FILLER 1</td>
<td>DS XL2</td>
<td>Reserved area of 2 bytes</td>
</tr>
<tr>
<td>FILLER 2</td>
<td>DS 4</td>
<td>Reserved full word</td>
</tr>
</tbody>
</table>

Table row header

Table row header information is a table attribute. It precedes the row descriptor block information on the initialization call.

The "table" described in this topic can be a physical table or a logical table. For instance, if you are using the user exit with NGT Unload, the row descriptor defines the output row. Because NGT Unload can create columns as part of the unload process, there is no physical table column. However, the table still has attributes, such as a name and a number of columns.

+0(4)

Eyecatcher value CDBX

+4(4)

Version number of structure passed on initialization call

The initial value of this field is V100.

+8(4)

Creator ID pointer, which is the pointer to the creator ID of the table

This can have a maximum length of 128 characters.
+12(4)

Name pointer, which is the pointer to the table name

This can have a maximum length of 128 characters.

+16(4)

ColCount

This is the number of columns in the row being defined.

+20(4)

MaxLen

This is the maximum output row length.

+24(1)

VarFlag

■ Y—for varying length rows.
■ N—for fixed length rows.

+25(1)

Unused

+26(8)

SSID DB2 subsystem or group attach name to which the ROWEXIT module is already connected from the invoking process

+34(2)

RDBLen

This is the row descriptor block length, which represents the length of each repeating section for each field or column in the row.

+36(4)

Unused

---

**Row descriptor block**

The row descriptor block (array) is passed as an input parameter during the initialization call.
The ROWEXIT module populates the row descriptor block with information before the initialization call. This information is maintained throughout the life of the module, and immediately follows the table row header block on the initialization function call. The row descriptor block structure is repeated for each field or column in the table row.

+0 COLUMN NAME POINTER

4-byte pointer which holds the address of the column name

POINTER

This name field is a character field with a varying size with a maximum of 128 characters. For derived fields (such as expressions), the names are in the form of "EXP1", "EXP2", and so on.

+4 COLUMN LENGTH

Column length value, which is a 4-byte field

For decimal values, it takes in the format PPSS, where PP = 2-byte precision and SS = 2-byte scale.

For varying columns, this is the maximum length allowed (not including the 2-byte LL field).

+8 COLUMN TYPE

This 8-byte field identifies the column type. Values are as follows:

BLOB

LOB field

CHAR

character field

CLOB

LOB field

DATE

Date field

DBCLOB

LOB field
DECIMAL

Decimal field

FLOAT

Floating numeric field

GRAPHIC

Graphic field

INTEGER

Integer field

LONGVARG

Long variable graphic field

LONGVAR

Long variable character field

SMALLINT

Small integer field

TIME

Time field

TIMESTAMP

Timestamp field

VARCHAR

Variable character field

VARG

Variable graphic field

ROWID

Identity column

+16 NULLABLE

This 1-byte field specifies if the field can be NULL.
- Y—this field can be NULL.
- N—this field cannot NULL.

+17 NULL CHARACTER LENGTH

Null field indicator length of 1-byte

+18 NULL CHARACTER

Specifies the column NULL character value, with a maximum length of 4 characters

+22 VARYING TYPE

This 1-byte field indicates if the field is varying (V) or fixed (F).

+23 Unused

5 free bytes reserved for future use
Index

+CHARFLDDLM 174
+CONNECTALL parameter 158
+DATEFMT 133
+FIELDSTEP 174
+MINROWS 73, 140, 154
+NUMFLDDLM 174
+TIMEFMT 151

A

abends
  recovery from 75
  using TERMINATE UTILITY for 56, 75
accessibility of objects 79
aliases, specifying in FROM option 89
ALL keyword
  of ON FAILURE option 75
alternate OBID 19, 25
application-period temporal tables
  CNTLCARDS DB2LOAD 54
ASC keyword 93, 145
ASCII 27, 130
  data translation 122
  option 51
ASCIIDEL 138, 157, 159, 161, 163, 167, 168
AUTO keyword 140
  of DIRECT option 59
automation control points 146
automation exit processing 177
AUTOTAG option 51
availability, data 79

B

BIGINT 175
BIGINT data type 106
binary data 110, 112
BINARII data type 110
BINARY VARYING data type 110
bitemporal tables
  CNTLCARDS DB2LOAD 54
BLANKPAD 52
BLOB 194
BLOB data type 112
BMCXCOPY table
  cabinet copies 70
BYPART option 53, 131
C

cabinet copies, unloading from 68
CASE construct 18
CCSID 27, 130, 131, 134, 153
CCSID option
  description 53
CCSIDs (code character set identifiers) 51, 61, 73, 84
CDBESTAE 193
CDBFCODE 193
CDBMESSAGE 193
CDBRETCODE 182, 193
CDBROWDISP 193
CDBROWPTR 193
CDBUSERTOK 193
CDBX 193
CHAR 194
CHAR data type 100
character constants
  translation 123
character data
  collating sequence 76
  conversions (translation) supported 122
  data types 100, 102
  translation 122
character field delimiter 157
CHARDEL 173
CHARFLDDLM 27, 174
cleaning up unused data sets 56
CLOB 194
CLOB data type 112
clone tables 89
clones 17
CLUSTER keyword 145
clustering key, ordering by 76
CNTLCARDS options
  DB2LOAD keyword 54
description 54
  inserting additional command options using the string parameter 54
CNTLDDN 28
CNTLDDN option
description 55
coded character set ID. See CCSID
ColCount 193
COLDEL 173
columns
timestamp, with precision 111
command options
  rules for syntax 36
command syntax 35
comparison operators 91
compression
dictionary, objects containing 70, 71
concatenated copy data sets 71
concurrency
  SHRLEVEL option 79
constants
  AUTOTAG option 51
  command 123
  comparison 123
  IN keyword 123
  LIKE keyword 123
  NULL keyword 93
  NULLCHAR option 73
  predicate block 92, 123
  RECORDID option 89
  with WHERE clause 92
contraction, during translation 123
control statements
generating 54
  product type specification 54
conventions
  rules for specifying command options 36
conventions, documentation 12
conversions
data type, allowable 113
ersors 59
formats supported 113
order of 124
role of DISCARDS option 59
rounding of values 105, 106
supported types 98
TRIM and TRUNCATE options 101, 110
copies, cabinet 68
copies, unloading from. See image copies, unloading from
COPY 17
copy data sets
  unloading from 68
copy registration
  cabinet copies 70
copy, unloading from 166
COPYDDN 24
COPYDDN option 132
COPYDSN 24, 132
creator ID pointer 193
CSV 168
CURRENT DATE option 93, 99
CURRENT TIME option 99
CURRENT TIMESTAMP option 93, 99

D
data
  availability 79
  conversion 113
  translation 122
data conversions
  allowable 113
  numeric considerations 121
  with respect to data translation 124
data sets
  cleaning up unused 56
  concatenated 71
  copy
    unloading 68
  deleting 56
  overriding default ddnames 85
data translation
  process 122
  specifying options 61
data types

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
BIGINT 106
BIGINT EXTERNAL 106
BINARY 110
BINARY VARYING 110
BLOB 112
CHAR 100
CLOB 112
conversions 113, 124
DATE 111
DATE EXTERNAL 111
date/time considerations 121
DBCLOB 112
DECFLOAT 109
DECIMAL 107
DECIMAL EXTERNAL 107
DECIMAL PACKED 107
DECIMAL ZONED 107
default formats 98
default lengths 113
description of each 99
EXTERNAL 121
FLOAT 108
FLOAT EXTERNAL 108
GRAPHIC 103
INTEGER 106
INTEGER EXTERNAL 106
keywords 99
numeric considerations 121
option (with SELECT) 98
SMALLINT 105
SMALLINT EXTERNAL 105
 specifying 98, 99
TIME 111
TIME EXTERNAL 111
TIMESTAMP 111
TIMESTAMP EXTERNAL 111
TIMESTAMP WITH TIME ZONE 111
TIMESTAMP WITH TIME ZONE EXTERNAL 111
VARBINARY 110
VARCHAR 102, 121
VARGRAPHIC 104
data-sorting indexes 76
date
data type conversions 113
data types 111
overriding DB2 date exit routine 55
 specifying current 93
DATE 194
DATE data type 111
DATE EXTERNAL data type 111
DATE FORMAT 159
DATEFMT option 55
DB2 134
control statements 55
DB2SQL keyword 140
DBADM 17
DBCLOB 194
DBCLOB data type 112
DBCS 27
DD name 146
DD statements
 NGTAUTO 33
 SYSCNTL 33
 SYSIN 33
 SYSPUNCH 33
 SYSREC 33
 UTLPARMS 33
ddnames
 specifying for the control statement data set 55
 specifying for the output data set 85
 with INFILE option 68, 71
DECFLOAT 175
DECFLOAT data type 109
DECIMAL 175, 194
decimal data type 107
decimal data, zoned 85, 107
DECPT 173
default time format 164
DEGREE 160
DELETEFILES option 56
deleting work files
 specifying 56
DELIMITED 28, 159, 167, 168, 173
delimited tokens 36
DESC keyword 93, 145
dictionaries, compression
 specifying 70, 71
differences between UNLOAD PLUS and NGT
Unload 30
DIGITS option 134
DIRECT option 134
DIRECT keyword 140
DIRECT option
EBCDIC 27
  data translation 122
  option 61
error handling
  using ON FAILURE to recover from abnormal termination 56, 75
ESTROWS option
description 94

F

FETCH, specifying multiple rows for one request 59
field separator 161
field specification options 96
fields
  binary 110, 112
  character 100
  data type specification 98
  fixed-length 64
  IF option 98
  LOB data 112
  mixed double-byte character set (DBCS) 100
  output record identification 96
  specification option, with SELECT 96
  TRIM and TRUNCATE options 101, 110
  VARBINARY 110
  VARCHAR 102
  with FORMAT BMCLOAD 96
FIELDSEP 27, 174
FILL option
  command 62
  of BIGINT data type 106
  of CHAR data type 100
  of DECIMAL data type 107
  of INTEGER data type 106
  of SMALLINT data type 105
  of VARCHAR data type 102
FILLER 1 193

AUTO 59
description 57
NO
  description 58
YES
description 58
  relation to AUTO option 59
view restrictions 89
discarded records 59
DISCARDS option 59
DISP 130, 132
disposition
  and deleting files 56
documentation information 11
double-byte character strings. See DBCS
drain
  specifying the number of retry attempts 78
  specifying the time between retry attempts 78
  timeout value 60
DRAIN_WAIT option 60
DSN1COPY 24
  compression dictionary limitation 71
  source of unload 68
DSNDB04 database name 50
DSNHDEC 130, 134, 153
default values at installation 61
DSNTIAUL 28, 159, 167, 168
DSNTIAUL format, output in 67
DSNTIAUL keyword of FORMAT option
description 67
  relation to FIXEDVARCHAR option 64
  relation to NULLCHAR option 73
DSNUM 151
DSNZPARMS, utility timeout value 60
DUMMY 28
dynamic SQL
  DIRECT option 58
  specifying 57
EUR 159, 164
EXCLUDE PART option 62
EXCLUDE PARTS option 134
exit routines
  discarded records, cause of 59
  expansion, during translation 123
EXTERNAL 159, 167, 168
external format, output in 66
EXTERNAL keyword
  of BIGINT data type 106
  of DECIMAL data type 107
  of FLOAT data type 108
  of FORMAT option 66
  of GRAPHIC data type 103
  of INTEGER data type 106
  of SMALLINT data type 105
externalizing pages for table spaces 80

E
FILLER 2 193
FILSZ 139
filtering partitions 63
FILTERPART option
description 63
fixed-length output records 64
FIXEDVARCHAR option
description 64
FLOAT 175, 194
FLOAT data type 108
floating-point data types 108, 109
FORCE option 66
FORMAT options
description 66
DSNTIAUL keyword 67
EXTERNAL keyword 66
FROM option 89
full image copies, unloading from 70
FULL option 71
FULL keyword
of INFILE ddname option 71
of INFILE IMAGECOPY option 70
FULL option 70

G
GDG 132
GENERATED ALWAYS columns 58
generating control statements
DB2 LOAD 54
GRAPHIC 194
GRAPHIC data type
allowable data type conversions 113
description 103

H
hash tables 17

I
IF option with SELECT 98
image copies, specifying 132
image copies, unloading from 68, 132, 151
cabinet copies 68
compression dictionary limitation 70, 71
different subsystems 71
full 70, 71
incremental 70, 71
INFILE option 68
inline 71
SHRLEVEL CHANGE 80
SHRLEVEL REFERENCE 79
incremental image copies
specifying 70, 71
INCREMENTAL keyword
of INFILE ddname option 71
of INFILE IMAGECOPY option 70
Index Term 194
indexes
ordering by 76
INFILE 166
INFILE option
description 68
inline image copies, unloading from 71
INTEGER 175, 194
INTEGER data type 106
INTO option
description 88
IRLMWAIT, used with drain timeout 60
ISO 159, 164

J
JIS 159, 164

K
keywords
FILL option
of SMALLINT data type 105
GRAPHIC data type 103
INFILE option 68
VALUE option 99

L
LCL 159
length option
with BINARY data type 110
with DATE EXTERNAL data type 111
with TIME EXTERNAL data type 111
with TIMESTAMP EXTERNAL data type 111
with VARBINARY data type 110
with VARCHAR data type 102
with VARGRAPHIC data type 104
limit key specifications in DDL 71
LIMIT option
description 72
LOAD DATA 175
LOAD DATA statement 167
load statement 138
load utilities, control statements 54, 88
LOB data 149, 166
data types 112
discards due to length, limiting 59
LOGICAL option 77, 95
logical partitions 77, 95
long object names 36
LONGVAR 194
LONGVARG 194
LRECL 24, 28

M
MaxLen 193
MAXROWS 19, 165
mixed data, translation 123
MIXED keyword
of CHAR data type 100
of VARCHAR data type 102
MODE option 140
multi-row fetch 59
multi-table table spaces 71
multiple ON FAILURE statements 75
multiple SELECT statements 71

N
NAME option 88
name pointer 193
names of objects, long 36
NGTAUTO 33, 177
NOFILTER option 145
nonpartitioned table spaces 70
NOPAD 26, 147, 169, 171
NOSUBS option 73
NULL 175

keyword of IF option 99
null indicator
NULLCHAR option 73
NULLTYPE option 74
when to use if column is nullable 98
NULL keyword
of VALUE option 99
NULLCHAR 27
NULLCHAR option 73
NULLSTRING 27, 174
NULLTYPE option 74
numeric data types, converting to external 62
numeric field delimiters 163

O
OBID 145, 171
OBID option 90
OBID, specifying
for INFILE ddname 71
with OBID option 90
object availability, SHRLEVEL option 79
object names, long 36
object names, specifying 37
ON FAILURE option 75
operators 91
OPTIONS option 94
options, command
field specification 96
ORDER BY option 93, 145
ORDER option
description 76
OUTDDN option 146
output
DSNTIAUL format 67
external format 66
format 66
records, tagging 51
specifying format 66
OUTPUT command 34, 132, 146, 150
output descriptors
length 102
output records
identifying 88
tagging 51, 89
overpunch values, assigning 85
PACKED keyword of DECIMAL data type 107
PAD 147, 171
padding, order of 124
PART option
description 77, 95
partition-by-growth table spaces 63
partitioned table spaces
  INFILE IMAGECOPY option 70
  partition-by-growth table spaces
    with FILTERPART 63
    with INFILE ddname 71
  range-partitioned 71
  specifying the partitions to be unloaded 77
with INFILE ddname 71
partitioning key, ordering by 76
partitions
  logical
    specifying 77, 95
  rebalanced 71
  rotated 71
  specifying 77, 95
partitions, excluding 62, 134
partitions, filtering 145
PARTS 154
PDS 132
PIT 149
point-in-time. See PIT
precision option
  of DECIMAL data type 107
  of DECIMAL data type 107
predicate
  of IF condition 99
publications, related 11

QUIESCE option 80

RECFM 24, 26, 28, 139, 169, 171
record identifiers (RIDs)
  specifying a value for 89
RECORDID option 89
recovery from abnormal termination 75
REFERENCE keyword of SHRLEVEL option 79
related publications 11
RETRY option 78
RETRY_DELAY option 78
retrying drain 78
return code value 180
routines, user exit
  discarded records, cause of 59
row disposition value 180
ROWID 194
rowset-positioning cursors 59
rowset, specifying 59
ROWSETSZ option 59
RTRIM option 96
rules for command syntax 36
RW 17
RW (read/write) status, SHRLEVEL CHANGE option 80

SBCS 27
scale, specifying on numeric fields 121
SELECT 17
SELECT option 86, 148
SELECT statement 86
SELECT statements
  multiple 71
semicolon specification when using DIRECT NO 96
SHRLEVEL CHANGE 149
SHRLEVEL option 79
  CHANGE keyword 80
  REFERENCE keyword 79
single-byte character strings. See SBCS
SMALLINT 175, 194
SMALLINT data type 105
sort work data sets (SORTWK)
  deleting 56
sorting
  ORDER option 76
SORTWK data sets
deleting 56
SPANNED option 149
spanned-record output 80
SQL, dynamic, DB2 58
SSID 193
ST 164
STMTDDN 28, 167
STMTDDN option 150
subset of rows, unloading 165
synonyms, specifying in FROM option 89
syntax
rules for specifying 36
syntax, command 35
SYSADM 17
SYSCNTL 33, 167
SYSCNTL data sets
type of control statements written to 54
SYSCOPY 24, 151
SYSCtrl 17
SYSIBM.SYSCOPY 151
SYSIBM.SYSCOPY table 70, 80
SYSIN 33
SYSOUT 28
SYSPUNCH 28, 33, 167
SYSPUNCH DD 150
SYSREC 33
SYSREC data sets
changing default name 85
deleting 56
SYSREC keyword of UNLOADDDN option 85
SYSRECO 146
SYSRED keyword of UNLOADDDN option 85
system-period temporal tables
CNTL CARDS DB2LOAD 54
SYSTEMPAGES NO, copies defined with 70, 71

T

table spaces
multi-table 71
partition-by-growth
with FILTERPART 63
with INFILE ddname 71
range-partitioned 71
unloading entire 50
tables
application-period temporal
cntl CARDS DB2LOAD 54
bitemporal
cntl CARDS DB2LOAD 54
clone 89
specifying 88, 89
system-period temporal
cntl CARDS DB2LOAD 54
temporal
cntl CARDS DB2LOAD 54
TABLESPACE option 50
tagging records
AUTOTAG option 51
RECORDID option 89
temporal tables
cntl CARDS DB2LOAD 54
termination, recovering from abnormal 75
time
CURRENT TIME option 99
field 111
overriding the default DB2 external format 81
TIME 194
TIME data type 111
time zone
data type 111
TIMEFMT option 81
timeout
specifying 60
specifying the number of retry attempts 78
specifying the time between retry attempts 78
timestamp
CURRENT TIMESTAMP option 93, 99
default DB2 external format, overriding 83
field 111
with time zone 111
TIMESTAMP 152
TIMESTAMP data type 111
TIMESTAMP WITH TIME ZONE data type 111
TIME 194
tokens in syntax, rules for 36
trailing blanks, removing
TRIM option 101, 103, 110
translation
command constants 122  
data 122  
order of 124  
processing 122  
restrictions 122  
specifying 61  
supported 122  
turning off substitution characters during translation between CCSIDs 73
TRIM option  
of BINARY data type 110  
of CHAR data type 101  
of VARCHAR data type 103
trimming options  
TRIM option  
of BINARY data type 110  
of CHAR data type 101  
of VARCHAR data type 110  
of VARCHAR data type 103
TRUNCATE option  
of BINARY data type 110  
of CHAR data type 101  
of GRAPHIC data type 103  
of VARCHAR data type 110  
of VARCHAR data type 103  
of VARGRAPHIC data type 104
truncating  
binary values 110  
character values 101  
graphic values 103
TSFMT option 83
UNLOAD PLUS syntax 128  
UNLOAD TABLESPACE 51  
UNLOAD DN option  
unloading from an image copy 151  
USA 159, 164
user exit routines  
discarded records, cause of 59
UTILITY keyword of ON FAILURE option 75  
UTILITYTERM keyword of ON FAILURE option 75
UTLPARMS 33
VALUE option 99  
VARBINARY data type 110  
VARCHAR 194  
VARCHAR columns 147  
VARCHAR data type 102, 121  
VarFlag 193  
VARG 194  
VARGRAPHIC data type 104, 121  
views, specifying 89
VSAM 132
WHERE clause 19, 53, 131  
condition of SELECT option 91
WK00001 18
work files  
deleting 56
worklist environment  
DELETFILES option 56
XML data  
discards due to length, limiting 59
XSUT0000 177
XSUTGLOB 177
XSUTSYIN 177
XSUTTERM 177
XSUTXERR 177
XULDDYNM 146

ULDPARMS 33, 133, 151
Unicode  
option to support 84  
supported translation 122  
translation process 122
UNICODE 27
UNICODE option 84
union in view 89
universal table spaces 63
UNLD 131
UNLD command 35
UNLOAD 167, 168
UNLOAD command 35, 50
UNLOAD PLUS 30
zeros
  with FILL option 62, 100
  with FIXEDVARCHAR option 64
  with TRIM option 110
  with TRUNCATE option 110
zoned decimal data

assigning overpunch values using
  ZONEDDECOVP option 85
data type 107
  ZONEDDECOVP option
  command 85