Contacting BMC Software

Several methods are available for contacting BMC Software.

You can access the BMC Software website at http://www.bmc.com. From this website, you can obtain information about the company, its products, corporate offices, special events, and career opportunities.

United States and Canada

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

Outside United States and Canada

<table>
<thead>
<tr>
<th>Telephone</th>
<th>Fax</th>
</tr>
</thead>
<tbody>
<tr>
<td>+01 713 918 8800</td>
<td>+01 713 918 8000</td>
</tr>
</tbody>
</table>
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.
# Table Of Contents

## Chapter 1
**Summary** .................................................................................................................. 1-1
1.1 Features ...................................................................................................................... 1-4
1.2 Usage Notes .............................................................................................................. 1-4
1.3 Design for Performance ......................................................................................... 1-5

## Chapter 2
**Input: DD Statements, Syntax, Keywords** .................................................................. 2-1
2.1 DD Statements ........................................................................................................... 2-1
2.2 SYsin Syntax Diagrams ............................................................................................ 2-3
2.2.1 REORG TABLESPACE Syntax ............................................................................. 2-3
2.2.2 REORG TABLESPACE Keywords ...................................................................... 2-5
2.2.3 REORG INDEX Syntax ....................................................................................... 2-12
2.2.4 REORG INDEX Keywords ................................................................................... 2-12
2.2.5 REORG APPLICATION Syntax ......................................................................... 2-13
2.2.6 REORG APPLICATION Keywords ..................................................................... 2-13

## Chapter 3
**Parameters** .................................................................................................................. 3-1
3.1 NGT Reorg Control Datasets ..................................................................................... 3-1
3.2 Parameter Overview .................................................................................................. 3-1
3.2.1 +ALLO .................................................................................................................. 3-3
3.2.2 +BEFOREACC(UT) ............................................................................................ 3-4
3.2.3 +FREEPAGE ......................................................................................................... 3-5
3.2.4 +INITWINDOW ..................................................................................................... 3-6
3.2.5 +IGNDEFFNO ....................................................................................................... 3-7
3.2.6 +LOB .................................................................................................................... 3-8
3.2.6 +LOBPCT .............................................................................................................. 3-9
3.2.7 +MAXTASKS ....................................................................................................... 3-10
3.2.8 +MINTSSIZE ....................................................................................................... 3-11
3.2.9 +PAGEUSE ............................................................................................................ 3-12
3.2.10 +PASSWORD ....................................................................................................... 3-13
3.2.11 +PBGCHKDICT ................................................................................................. 3-14
3.2.12 +PCTFREE ........................................................................................................... 3-15
3.2.13 +RUNSTATS ....................................................................................................... 3-16
3.2.14 +VOLUMES ....................................................................................................... 3-18

## Chapter 4
**Examples** ..................................................................................................................... 4-1
4.1.1 Reorganize a table space ...................................................................................... 4-1
Table Of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1.2</td>
<td>Reorganize specific parts of a partitioned table space</td>
<td>4-1</td>
</tr>
<tr>
<td>4.1.3</td>
<td>Reorganize specific logical parts of a partitioned table space</td>
<td>4-1</td>
</tr>
<tr>
<td>4.1.4</td>
<td>Reorganize a table space and generate an image copy</td>
<td>4-2</td>
</tr>
<tr>
<td>4.1.5</td>
<td>Reorganize a table space, sending discards to a dataset</td>
<td>4-2</td>
</tr>
<tr>
<td>4.1.6</td>
<td>Reorganize a set of table spaces, excluding certain table spaces</td>
<td>4-2</td>
</tr>
<tr>
<td>4.2</td>
<td>NGT Reorg Index</td>
<td>4-2</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Reorganize an index</td>
<td>4-2</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Reorganize specific parts of an index</td>
<td>4-2</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Reorganize a set of indexes excluding certain indexes</td>
<td>4-3</td>
</tr>
<tr>
<td>5</td>
<td>Chapter 5 Summary Reports</td>
<td>5-1</td>
</tr>
<tr>
<td>5.1</td>
<td>Internal IDS Report</td>
<td>5-1</td>
</tr>
<tr>
<td>5.2</td>
<td>Object Space Report</td>
<td>5-2</td>
</tr>
<tr>
<td>5.3</td>
<td>Before-After Report (BF AF)</td>
<td>5-2</td>
</tr>
<tr>
<td>6</td>
<td>Chapter 6 Relevant Automation</td>
<td>6-1</td>
</tr>
<tr>
<td>6.1</td>
<td>XRRGINIT (Reorg Initialization)</td>
<td>6-1</td>
</tr>
<tr>
<td>6.2</td>
<td>XRRGTERM (REORG Completion)</td>
<td>6-1</td>
</tr>
<tr>
<td>6.3</td>
<td>XRRGRWRO</td>
<td>6-1</td>
</tr>
<tr>
<td>6.4</td>
<td>General Utility Automation Control Points</td>
<td>6-2</td>
</tr>
<tr>
<td>7</td>
<td>Chapter 7 Technical Details</td>
<td>7-1</td>
</tr>
<tr>
<td>7.1</td>
<td>Reasons to Reorg</td>
<td>7-1</td>
</tr>
<tr>
<td>7.1.1</td>
<td>Reasons to Not Reorg</td>
<td>7-2</td>
</tr>
<tr>
<td>7.1.2</td>
<td>Index Reorg Advantages</td>
<td>7-2</td>
</tr>
<tr>
<td>7.1.3</td>
<td>Reorganizing Partition By Growth (PBG) table spaces</td>
<td>7-2</td>
</tr>
<tr>
<td>7.2</td>
<td>Fine Tuning for the NGT REORG Utility</td>
<td>7-3</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Automatic Restart</td>
<td>7-3</td>
</tr>
<tr>
<td>7.2.2</td>
<td>NGT Reorg vs. IBM Reorg Restart</td>
<td>7-3</td>
</tr>
<tr>
<td>7.2.3</td>
<td>Specifying a Utility ID</td>
<td>7-3</td>
</tr>
<tr>
<td>7.2.4</td>
<td>The Checkpoint Dataset</td>
<td>7-4</td>
</tr>
<tr>
<td>7.3</td>
<td>Specifying Free Space</td>
<td>7-4</td>
</tr>
<tr>
<td>7.4</td>
<td>Processing DB2 objects that are Reorg Pending</td>
<td>7-5</td>
</tr>
<tr>
<td>7.4.1</td>
<td>Minimize down-time due to REORP</td>
<td>7-5</td>
</tr>
<tr>
<td>7.4.2</td>
<td>DDL That cause REORP status</td>
<td>7-5</td>
</tr>
<tr>
<td>7.5</td>
<td>Processing DB2 indexes that are Rebuild Pending</td>
<td>7-5</td>
</tr>
<tr>
<td>7.5.1</td>
<td>Minimize down-time due to RBDP or PSRBD</td>
<td>7-6</td>
</tr>
<tr>
<td>7.5.2</td>
<td>DDL That cause RBDP or PSRBD status</td>
<td>7-6</td>
</tr>
<tr>
<td>7.6</td>
<td>Materializing On-Line Schema changes</td>
<td>7-7</td>
</tr>
<tr>
<td>7.7</td>
<td>Processing table spaces with LOBs</td>
<td>7-8</td>
</tr>
</tbody>
</table>
Table Of Contents

Index ........................................................................................................................................X-1
### Chapter 1

## Summary

**BMC Next Generation Technology Reorg**

NGT Reorg is a utility for reorganizing database objects.

<table>
<thead>
<tr>
<th>When To Use It</th>
<th>When you need to reorganize database objects without stopping the table spaces.</th>
</tr>
</thead>
</table>
| Authorizations Required | **Database:** REORG authority  
**User:**  
SYSADM authority  
DBADM authority |
| Restrictions | **DDL Changes.** NGT Reorg will terminate if a table is added to or dropped from the table space during the reorganization process. Rerun the utility.  
**Mass Delete.** NGT Reorg will terminate if a table is mass deleted from a segmented table space during the reorganization process. Rerun the utility.  
**Empty TS with NPI clustering IX.** NGT Reorg will skip a table space if it is empty and is clustered by an NPI.  
**V10 Hashed table space.** The initial delivery of NGT Reorg does not process Hashed table spaces. They are detected and skipped.  
**Cloned Tables.** The initial delivery of NGT Reorg does not process table spaces with cloned tables. They are detected and skipped.  
**XML Auxiliary table space.** The initial delivery of NGT Reorg does not process XML Auxiliary table spaces. They are detected and skipped.  
**UTS PBG table space with LOB columns.** NGT Reorg currently does not process PBG table spaces with LOB columns. They are detected and skipped.  
**Auxiliary storage requirements for a PBG table space Reorg.** To run an NGT Reorg of a Partitioned by Growth table space, your system must have at least 2GB of auxiliary storage. |
storage; regardless of whether 2G of memory will ultimately be needed or not. 

**DSSIZE** greater than 64GB is currently not supported. 

**Table space with an Added Identity column.** The initial delivery of NGT Reorg does not process table spaces that have been altered to add an identity column. Table spaces with existing identity columns are processed.

<table>
<thead>
<tr>
<th>Statuses</th>
<th>DB2 States</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Table space: RW, non-restricted states only</td>
</tr>
</tbody>
</table>
NGT Reorg Index

NGT Reorg Index is a utility for reorganizing indexes.

<table>
<thead>
<tr>
<th>When To Use It</th>
<th>To reorganize database indexes without stopping the database.</th>
</tr>
</thead>
</table>
| **Authorizations Required** | **Database:** DBADM authority  
**Administrator:** SYSADM authority |
| **Restrictions** | **DDL Changes.** NGT Reorg Index will terminate if a table is added to or dropped from the table space during the reorganization process. Rerun the utility.  
**Mass Delete.** NGT Reorg Index will terminate if a table is mass deleted from a segmented table space during the reorganization process. Rerun the utility.  
**Rebuild Pending.** NGT Reorg Index will terminate if an index is in Rebuild Pending (RBDP or PSRBD) status. Run REBUILD INDEX to make the index RW. |
| **Statuses** | **DB2 States**  
Index: RW, non-restricted states only |
1.1 Features

**Runstats Updated.** NGT Reorg can optionally update the DB2 catalog with the RUNSTATS values calculated at the end of the reorganization.

If a subset of partitions are reorganized, statistics are updated for those partitions only and composite statistics are recomputed and updated.

**Concurrent Copy Support.** "Concurrent" refers to copies made at the same time as a LOAD or REORG.

NGT Reorg will create DB2 Image Copies during a Reorg on any table space. The copies are made concurrently with the writing to the table space; a separate run is not required.

Concurrent Copy utilizes DASD or VTS. If the copies are to DASD, they are DSN1COPY compatible. For Cartridge tapes needed for offsite storage: (1) the Recovery Site copy should go to DASD that is immediately HSM migrated to tape. This makes efficient use of tape and shortens recovery time by allowing for recall prior to starting the recovery. (2) Use a VTS that has external tapes that can be taken offsite and brought back. This, too, makes efficient use of tapes and buffers them to DASD for faster recovery times.

**SYSCOPY Posting.** NGT Reorg will post the Reorgs (and all concurrent copies) in the DB2 SYSIBM.SYSCOPY Catalog table. The Reorg is posted as a LOG(NO) Reorg.

**Delete During Reorg.** NGT Reorg can delete rows from the table during Reorg if a DISCARD clause is specified. The DISCARD clause uses an SQL-like WHERE clause that specifies which rows are to be deleted during the REORG. These rows are not logged, but can be archived to a dataset. This Discard dataset will be in DB2 REORG UNLOAD format.

DISCARD is not allowed on a System-maintained temporal table spaces when versioning is active. On such table spaces deleted rows go to the history table.

The WHERE clause can use Boolean logic, LIKE and IN predicates, subselects in an IN clause, and all DATE, TIME, and TIMESTAMP functions. For more information on the WHERE clause, including syntax and restrictions, see the *BMC Next Generation Technology General User Guide*.

1.2 Usage Notes

**Copy Servers.** NGT Reorg, for greater efficiency, may initiate an extra copy server. This server job is solely for the purpose of producing image copies from objects that have been already reorganized and completed.

**Row Counts May Occasionally Vary.** Users may observe that, on occasion, the number of keys reported before and after the Reorg process may differ. This may be due to deletions performed before or during the Reorg process and is not indicative of a program error or data integrity issue.
1.3 **Design for Performance**

**DSSIZE.** NGT Reorg will process the legacy Segmented table spaces with sequential reads and sequential writes. Segmented table spaces should be converted to PBG UTS. If that PBG is one 64G partition, Reorg will still be limited to sequential read and sequential write. However if this PBG were 8 8G parts instead of 1 64G part, the Reorg will be *much* faster. NGT Reorg would not only read the 86 partitions in parallel, it will write the 8 partitions in parallel too.

With all partitioned table spaces, having DSSIZE greater than 32G does not increase the ultimate size of the table space; it only reduces the number of parts and the opportunity for parallelism.
Input: DD Statements, Syntax, Keywords

This chapter covers the DD statements, SYSIN input (statement syntax), keywords and parameters for this utility.

2.1 DD Statements

SYSIN
The only required DD statement for use with this utility is SYSIN. Specify utility statements as follows:

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

Statement syntax is covered in Sec. 2.2, "SYSIN Syntax Diagram".

RRGPARMS
RRGPARMS is only required to override the program’s parameters specified at installation time. They may be specified in a dataset or in the JCL, as in:

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

or

```
//RRGPARMS DD *
```

RRGPARMX
RRGPARMX is similar to RRGPARMS, except parameters specified in this section refer only to index processing:

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

or

```
//RRGPARMX DD *
```

UTLPARMS
UTLPARMS is only required to override the NGT Utilities global parameters specified at installation time. They may be specified in a dataset or in the JCL, as in:

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

```plaintext
//UTLPARMS DD *
```

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

**RUNSTATS**

RUNSTATS is an optional DD statement to which reports – Runstats – are written. This DD statement can be sent to an output file or to a SYSOUT output class as in:

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

or

```plaintext
//RUNSTATS DD SYSOUT=* 
```

**CDBEXEC**

To use automation control points, specify a CDBEXEC DD statement. This statement should reference a partitioned dataset containing the NGT automation control points needed for processing. An example of this DD statement is shown here:

```plaintext
//CDBEXEC DD DISP=SHR,DSN=NGT.CDBEXEC
```

For more information on automation exits, see the manual *BMC Next Generation Technology Automation Reference Guide*. 
2.2 **SYSIN Syntax Diagrams**

2.2.1 **REORG TABLESPACE Syntax**

```
REORG TABLESPACE dbname.spacename
  dbname.spacename
  dbname.spacename
  dbname.spacename

EXCLUDE TABLESPACE dbname.spacename
  dbname.spacename
  dbname.spacename

PREFORMAT
  PREFORMAT
  PREFORMAT

REBUILDICT
  REBUILDICT
  REBUILDICT

NOCLIX
  NOCLIX
  NOCLIX

LOBRECLAIMPERCENT integer
  LOBRECLAIMPERCENT integer
  LOBRECLAIMPERCENT integer

REPORTONLY
  REPORTONLY
  REPORTONLY

PART integer
  PART integer
  PART integer

PARTS integer
  PARTS integer
  PARTS integer

LPART integer
  LPART integer
  LPART integer

LPARTS integer
  LPARTS integer
  LPARTS integer

COPYDDN(SYSCOPY lp1,lb1)
  COPYDDN(SYSCOPY lp1,lb1)
  COPYDDN(SYSCOPY lp1,lb1)

RECOVERYDDN(rp1,rb1)
  RECOVERYDDN(rp1,rb1)
  RECOVERYDDN(rp1,rb1)

MASSDELETE
  MASSDELETE
  MASSDELETE

PART integer
  PART integer
  PART integer

PARTS integer
  PARTS integer
  PARTS integer

LPARTS integer
  LPARTS integer
  LPARTS integer

REBUILDDICT
  REBUILDDICT
  REBUILDDICT

REPORTONLY
  REPORTONLY
  REPORTONLY

LOBRECLAIMPERCENT integer
  LOBRECLAIMPERCENT integer
  LOBRECLAIMPERCENT integer

NOCLIX
  NOCLIX
  NOCLIX

PREFORMAT
  PREFORMAT
  PREFORMAT

REBUILDICT
  REBUILDICT
  REBUILDICT

COPYDDN(SYSCOPY lp1,lb1)
  COPYDDN(SYSCOPY lp1,lb1)
  COPYDDN(SYSCOPY lp1,lb1)

RECOVERYDDN(rp1,rb1)
  RECOVERYDDN(rp1,rb1)
  RECOVERYDDN(rp1,rb1)

MASSDELETE
  MASSDELETE
  MASSDELETE

PART integer
  PART integer
  PART integer

PARTS integer
  PARTS integer
  PARTS integer

LPARTS integer
  LPARTS integer
  LPARTS integer

COPYDDN(SYSCOPY lp1,lb1)
  COPYDDN(SYSCOPY lp1,lb1)
  COPYDDN(SYSCOPY lp1,lb1)

RECOVERYDDN(rp1,rb1)
  RECOVERYDDN(rp1,rb1)
  RECOVERYDDN(rp1,rb1)

MASSDELETE
  MASSDELETE
  MASSDELETE

PART integer
  PART integer
  PART integer

PARTS integer
  PARTS integer
  PARTS integer

LPARTS integer
  LPARTS integer
  LPARTS integer

RBALRSN_CONVERSION BASIC
  RBALRSN_CONVERSION BASIC
  RBALRSN_CONVERSION BASIC

EXTENDED
  EXTENDED
  EXTENDED

NONE
  NONE
  NONE
```

**copy-spec**

```
COPYDDN(SYSCOPY lp1,lb1)
  COPYDDN(SYSCOPY lp1,lb1)
  COPYDDN(SYSCOPY lp1,lb1)

RECOVERYDDN(rp1,rb1)
  RECOVERYDDN(rp1,rb1)
  RECOVERYDDN(rp1,rb1)
```

**massdelete-spec**

```
MASSDELETE
  MASSDELETE
  MASSDELETE

PART integer
  PART integer
  PART integer

PARTS integer
  PARTS integer
  PARTS integer

LPARTS integer
  LPARTS integer
  LPARTS integer

RBALRSN_CONVERSION BASIC
  RBALRSN_CONVERSION BASIC
  RBALRSN_CONVERSION BASIC

EXTENDED
  EXTENDED
  EXTENDED

NONE
  NONE
  NONE
```
NOTE:
As of Version 9 of NGT Reorg, **simple, multi-table table spaces are NOT supported.** A reorg may be run on a simple table space containing exactly one table, but may **not** be run on a simple table space with **more than one** table. Simple, multi-table table spaces may not be created in DB2 Version 9.
2.2.2 REORG TABLESPACE Keywords

dbname.spacename
If the dbname is not specified, the default name is DSNDB04.

dbmask.spacemask
The dbmask and spacemask parameters employ DB2 SQL wildcards to reference any number of different objects. DB2 SQL wildcards are used precisely as they are used in a DB2 WHERE clause after the LIKE keyword. Any combination of characters, "%" sign, or "." sign can be used in either name. An SQL query will be constructed and issued to obtain the names of all table spaces that match the specified dbmask and spacemask. A line will print on SYSPRINT and SUMMARY for each object found.

To reorganize all the table spaces in a database, specify dbname.%. For example,

```
REORG TABLESPACE BIGDBASE.A%
```

will reorganize every table space in the data base BIGDBASE whose space name begins with the character "A".

PART integer
This keyword specifies that only a single physical partition is to be reorganized. The PART keyword is used for a range of partitions.

PARTS parts-list
This keyword specifies that only the physical partitions in the specified list of partitions are to be reorganized. The values in the range must be separated by a comma and one or more of the values can be a range of contiguous parts from x to y specified as x:y where the second value must be smaller than the first.

NOTE:
This option does not support use of wildcards.

LPART integer
This keyword specifies that only a single logical partition is to be reorganized. (Logical partitions become different from the Physical partitions by the use of DB2's ROTATE feature.)

LPARTS parts-list
This keyword specifies that only the physical partitions in the specified list of partitions are to be reorganized. The values in the range must be separated by a comma and one or more of the values can be a range of contiguous parts from x to y specified as x:y where the second value must be smaller than the first.

NOTE:
This option does not support use of wild cards.

☑️ NOTE:
The PART keywords are ignored when reorganizing a PBG. All PBG partitions are reorganized simultaneously.

MASSDELETE
This keyword specifies that all rows of the partitions specified by the PART, PARTS, LPART or LPARTS keywords will be deleted from the table. In order for the MASSDELETE keyword to be accepted, the following must be true:

- DISCARD keyword must be absent.
- PART or LPART keyword must be specified with a single partition number or
- PARTS or LPARTS keywords must be specified with a range of partition numbers.

**IMPORTANT**

The part range specified for MASSDELETE must be a subset of the partitions specified for a Reorg. For example, if a 32-part table space is to be processed, and parts 18 to 24 have been specified for Reorg, any of the following part ranges would be valid for MASSDELETE: parts 17 to 19, parts 20 to 23, parts 21 & 22 (all of these are subsets of parts 18 to 24). On the other hand, parts 23 to 26 could not be used for MASSDELETE since parts 25 and 26 are outside the original range of parts 18 to 24.

In one input stream, only one REORG TABLESPACE utility statement for a table space may be specified. (Multiple statements referring to the same table space will NOT be grouped under one unit of work and will result in processing the object multiple times.)

When a partition is subject to MASSDELETE, the compression dictionary will be maintained or removed based on the COMPRESS catalog flag. If the Compress flag is on but the partition does not have a dictionary, a dictionary will be built from the data currently in the partition.

Any update or delete to a row in a partition subject to MASSDELETE during the reorg will result in the utility abending. Newly inserted rows will not be subject to MASSDELETE processing.

**DSNUM ALL**

The presence of this keyword causes all parts of a Partition By Growth (PBG) to be reorged at once and rows to be relocated between partitions if necessary and new partitions to be added if needed. DSNUM ALL, while still permitted, is no longer necessary as NGTReorg always processes all PBG parts.

**ROWFORMAT**

Used when performing a Reorg to convert the row format.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRF</td>
<td>Convert from RRF to BRF</td>
</tr>
<tr>
<td>RRF</td>
<td>Convert from BRF to RRF</td>
</tr>
</tbody>
</table>

**AUX**

Used to trigger the BMC Next Generation Technology LOBMaster product to Check and Reorg Base and LOB table spaces together online. Requires a NGT LOBMaster license; see the [BMC Next Generation Technology LOBMaster for DB2 for z/OS Reference Manual](#) for more details.

**NOCLIX**

This keyword is only valid with segmented non-partitioned table spaces or PBGs. This keyword causes Reorg to ignore the clustering index and process the table space as if there were no clustering index. This will cause Reorg to reclaim space only.

**MAXDISORG**

integer
This keyword prevents unexpected expensive reorgs of very large, highly disorganized PBG table spaces. Integer is a percentage of the rows that are disorganized. This keyword is ignored if the NOCLIX keyword is specified. The default MAXDISORIG value is 50, or 50% disorganized.

See Reorganizing Partition By Growth (PBG) table spaces in Chapter 7 of this manual for more details about reorganizing PBGs.

**PREFORMAT**

The presence of this keyword causes the table space pages to be preformatted when they are allocated. If PREFORMAT is specified, all pages up to the high allocated RBA are preformatted. Preformat consists of writing zero pages from the high used RBA to the high allocated RBA.

**LOBRECLAIMPERCENT integer**

Specifies the percentage of savings that should be attained by Reorg LOB. If the savings cannot be met, then the Reorg will be skipped. The savings is calculated by subtracting the number of pages needed for the output dataset from the current HURBA. Values can be between 0 and 99. If it is not specified then the default value is 20. Specifying 0 will force the Reorg to run. This keyword can be overridden by Reorg parm +LOBPCT; see Section 3 below for details.

**RBALRSN_CONVERSION**

Used to convert the DB2 RBAs and LRSNs between basic (6 byte) and extended (10 byte) format. The default behavior depends on the UTILITY_OBJECT_CONVERSION subsystem parameter, see Note below.

**BASIC**

Convert from extended to basic format.
If BASIC is specified and the UTILITY_OBJECT_CONVERSION subsystem parameter is set to NOBASIC; the object fails reorg.

**EXTENDED**

Convert from basic to extended format.
If EXTENDED is specified and the UTILITY_OBJECT_CONVERSION subsystem parameter is set to BASIC; the object fails reorg.

**NONE**

Do no conversions.
If NONE is specified and the UTILITY_OBJECT_CONVERSION subsystem parameter is set to NOBASIC and the object is in basic format; the object fails reorg.

**NOTE:**

If RBALRSN_CONVERSION is not specified and UTILITY_OBJECT_CONVERSION is set to BASIC, then RBALRSN_CONVERSION defaults to BASIC; if UTILITY_OBJECT_CONVERSION is set to NOBASIC or EXTENDED then RBALRSN defaults to EXTENDED.

**COPYDDN (lp1, lb1)**

Used to specify the desired local copy destinations. The first parameter specifies a Local Primary (LP) and the second parameter, a Local Backup (LB).

Starting with NGT Utilities Version 9.0.6, these parameters do not specify a DD name; they only signify the desire to produce a copy to this destination. The actual UNITNAME and options for the copy destinations are defined by the CDBTAPE statement or by NGT Automation. See the BMC Next Generation Technology General User Guide and the BMC Next Generation Technology Automation Reference Guide for further details.
**IBM SHRLEVEL CHANGE** Reorg requires that a COPY be done with the Reorg. BMC NGT is following IBM's standards. If this clause is not specified, the default is to produce a Local Primary (LP1) copy only.

Note that the parenthesis are optional if only one destination name is specified, e.g., COPYDDN LP1. The parentheses are required if two DD names are specified.

It is required that the NGT Automation facility control point **XCPYDYNM** be available to provide the image copy dataset name for each image copy.

**RECOVERYDDN (rp1, rb1)**
This keyword performs the same function as it does on the IBM COPY statement.

Used to specify the desired local copy destinations. The first parameter specifies a Remote Primary (RP) and the second parameter specifies a Remote Backup (RB). Starting with NGT Utilities version 9.0.6, these parameters do not specify a DD name, they only signify the desire to produce a copy to this destination. The actual UNITNAME and options for the copy destinations is defined by the **CDBTAPE** statement or by NGT Automation. See the **BMC Next Generation Technology General User Guide** and the **BMC Next Generation Technology Automation Reference Guide** for further details.

Note that the parentheses are optional if only one destination name is specified, e.g., COPYDDN LP1. The parenthesis are required if two DD names are specified.

It is required that the NGT Automation Facility control point **XCPYDYNM** be available to provide the Image Copy dataset name for each Copy.

**FLASHCOPY YES**
To request a Flashcopy from Reorg

See also NGT Global parameter +FLASHCOPY for options for what action to take if the volumes for the copy do not support Flashcopy.

**NOTE:**
Flashcopy datasets produced by Reorg are VSAM. Because of this the copy dataset cannot be a GDG dataset. Also, there will be a Cluster and Data entry so the DSNAME must leave two bytes for the “.D” to be added. XCPYDYNM can produce a different name for Flashcopy datasets.

**REBUILDICT**
**REBUILDICT** tells processing to rebuild the data dictionary.

**REPORTONLY**
Reorg will generate some SUMMARY reporting then stop. The reorganization of the object will NOT take place.

**DISCARDDN** ddbname
The presence of this keyword signals that all deleted rows are to be copied to a sequential file. The ddbname of this file is named in the value that follows the DISCARDDN keyword. DCB characteristics will be set internally. Each deleted row is written as a separate record in the file. The format is the DB2 REORG UNLOAD format, suitable for being loaded into a similar table using the LOAD DATA ... FORMAT UNLOAD statement. See the **BMC Next Generation Technology Load for DB2 for z/OS Reference Manual** for details.
Do not use the RLSE option on the SPACE parameter in this DD statement. This can cause a B37 when the dataset is reopened for a partitions discards.

If discard processing is being used (the DISCARD FROM TABLE clause), but neither the DISCARDDN keyword nor a //SYSDISC DD are present, then any rows discarded will be permanently lost, regardless of whether or not an XULDDYNM automation exit is present in the automation exit library (CDBEXEC). These rows are NOT sent to any dataset, and no dynamic dataset is automatically allocated.

For details on XULDDYNM, see the BMC Next Generation Technology Automation Reference Guide.

If either SYSDISC or DISCARDDN are specified then the discarded rows will be sent to the datasets specified, or if no dataset is defined and DISCARDDN is specified, then the dataset will be dynamically allocated using XULDDYNM (the dataset name is set with the dname variable).

If BOTH a SYSDISC DD and the DISCARDDN keyword are specified, then the discarded rows will only be sent to the dynamically-allocated dataset (with the dname set in the XULDDYNM automation exit), and not to any dataset associated with the SYSDISC DD.

**NOTE:**
When coding a Discard dataset for partitioned tables, be sure to specify MOD,CATLG,DELETE. Coding NEW,CATLG,DELETE will result in only the last parts discards being in the discard dataset.

**FORMAT**

FORMAT specifies the format of the Discard dataset. The DISCARDDN must be specified to use this option. Valid operands for the FORMAT keyword are:

**DSNTIAUL**

The output of the discarded rows will be in DSNTIAUL, or IBM's DB2 internal format. This can be abbreviated as DSN.

For the DSNTIAUL format only: If more than one table is specified on a DISCARD clause, then the output file will contain a 2-byte OBID value at the beginning of each row. If only one table is specified on a DISCARD clause, this 2-byte OBID will be absent.

**NOTE:**
This is a change since V7.4.0. Prior to DB2 V8 format DSNTIAUL discarded rows were prefixed with the table name. Since DB2 V8 the length of the table name are much longer and are now prefixed with the table OBID.

**UNLOAD**

Default. The output of the discarded rows will be in UNLOAD format. This can be abbreviated as ULD.

**EXTERNAL**

The output of the discarded rows will be in EXTERNAL format. This can be abbreviated as EXT.
REPORT
The output of the discarded rows will be in REPORT format. This can be abbreviated as REP.

ASCIIDEL
The output of the discarded rows will be in ASCIIDEL (ASCII, with values delimited by double quotes) format. This can be abbreviated as ASC.

PAD
The PAD option will cause the output file to pad VARCHAR columns to their full length. PAD is not valid for the following formats: EXTERNAL, REPORT, ASCIIDEL.

NOPAD
The NOPAD option will cause the output file not to pad VARCHAR columns to their full length – that is, to the actual length of the data only, minus any trailing blanks. NOPAD is not valid for the following formats: EXTERNAL, REPORT, ASCIIDEL.

If no FORMAT keyword is specified, the default format is UNLOAD. SYSPUNCH output will not be written if FORMAT UNLOAD is specified.

EXCLUDE
This keyword is used to exclude table spaces from being reorged, as in this example:

```sql
REORG TABLESPACE ADMINDB.CAR%
   EXCLUDE TABLESPACE ADMINDB.CARMIDSZ
   EXCLUDE TABLESPACE ADMINDB.CARLUXRY
   EXCLUDE TABLESPACE ADMINDB.CARSUVS
```

DISCARD FROM TABLE
If present, these keywords must appear last on the REORG statement. It specifies that rows are to be deleted as part of the Reorg process. DISCARD is not allowed on a System-maintained temporal table spaces when versioning is active. On such table spaces deleted rows go to the history table.

If the table space contains multiple tables, the FROM TABLE keywords (along with the table specified by creator.table) can be repeated. No comma separates the clauses, as in this example:

```sql
REORG TABLESPACE A.B DISCARDDN DELETES
   DISCARD FROM TABLE A.TABLE1 WHERE (COL01 = 'M')
   FROM TABLE A.TABLE2 WHERE (CUSTNO IN
   (SELECT CUSTOMER FROM A.TABLE3 WHERE INVOICE_DATE <
   CURRENT DATE - 30 DAYS))
```

creator.table
Names the table whose rows are to be deleted.

WHERE
This keyword (or its equivalent WHEN) begins the specification of an SQL WHERE clause.
The WHERE clause must be enclosed in parentheses.
The DISCARD option cannot be used with wildcards.

If DISCARD and RTMan are used together, rows will only be discarded from the partitions chosen by RTS for reorganization.

The DISCARD option cannot be used with wildcards.

See the chapter on WHERE clause support in the BMC Next Generation Technology General User Guide for information on NGT WHERE clause options. The most efficient WHERE clauses are the ones that NGT utilities can process natively without sending the SQL to DB2.

NGT WHERE also supports WHERE column IN FILE(ddname) which can be very efficient and useful when discarding the same keyed rows from multiple tables. The file specified in the WHERE clause is allocated to the Master job so this job must run with +SVRMODE(INTERNAL).

If //SYSPUNCH DD exists, the LOAD statement for the discarded rows will be written to the data set. In addition, the FORMAT keyword must be specified. (If FORMAT UNLOAD is selected, though, a LOAD statement will not be written to SYSPUNCH, although discarded rows will still be written.)
2.2.3 **REORG INDEX Syntax**

```plaintext
REORG INDEX creator.indexname
    INDEXES crmask.indexmask
    TABLESPACE dbname.spacemask
    INDEXSPACE dbname.indexpsa

PART integer
    LPARTS integer

INDEX creator.indexname
    Names the index to be reorganized.

INDEXES crmask.indexmask
    A set of indexes may be specified for processing by using the INDEXES keyword along with a naming mask containing wildcard characters. For details on the use of wildcards, see the *BMC Next Generation Technology General User Guide*.

INDEXSPACE dbname.indexpsa
    Names the indexspace to be reorganized.

INDEXSPACE dbname.ixspacemask
    A set of indexspaces may be specified for processing by using the INDEXSPACE keyword along with a naming mask containing wildcard characters. For details on the use of wildcards, see the *BMC Next Generation Technology General User Guide*.

PART integer
    This keyword specifies that only a single physical partition of an index is to be reorganized. The PART keyword is used for a range of partitions.

PARTS range
    This keyword specifies that only the physical partitions in the specified range of partitions are to be reorganized. The values in the range must be separated by a colon and the second value must not be smaller than the first.

NOTE:
    This option does not support use of wildcards.

LPART integer
    This keyword specifies that only a single *logical* partition of an index is to be reorganized. (Logical partitions become different from Physical partitions by the use of DB2's ROTATE feature.)

LPARTS range
```
This keyword specifies that only the logical partitions in the specified range of partitions are to be reorganized. The values in the range must be separated by a colon and the second value must not be smaller than the first.

**NOTE:**
This option does not support use of wild cards.

**REPORTONLY**
Reorg will generate some SUMMARY reporting then stop. The reorganization of the object will NOT take place.

**EXCLUDE**
This keyword is used to exclude indexes from being REORGed. Note that both of the following usages are valid:

```
REORG INDEXES TABLESPACE SOMEDB.% EXCLUDE TABLESPACE SOMEDB.ACCTS
REORG INDEXES TABLESPACE SOMEDB.% EXCLUDE INDEX SOMEDB.IXACCTS
```

### 2.2.5 REORG APPLICATION Syntax

```
REORG APPLICATION applname
  PREFORMAT
  EXCLUDE TABLESPACE dbname.spacename
    dbmask.spacemask
```

### 2.2.6 REORG APPLICATION Keywords

An application is, simply put, a collection of table spaces grouped as one object with one common name. The term derives from the fact that many software packages – such as information systems that manage data spread across a large number of tables – are the back end for computer applications such as online web applications, desktop programs that connect to DB2, and the like. NGT conceptualizes an application as a convenient control unit for managing entire sets of table spaces. For more information on applications, see the *BMC Next Generation Technology General User Guide*.

**EXCLUDE**
This keyword is used to exclude table spaces from being reorged, as in this example:

```
REORG APPLICATION CARS868
  EXCLUDE TABLESPACE ADMINDB.CARMIDSZ
  EXCLUDE TABLESPACE ADMINDB.CARLUXRY
  EXCLUDE TABLESPACE ADMINDB.CARSUVS
```

**PREFORMAT**
The presence of this keyword causes the table space pages to be preformatted when they are allocated. If PREFORMAT is specified, all pages up to the high allocated RBA is preformatted. Preformat consists of writing zero pages from the high used RBA to the high allocated RBA.
Input: DD Statements, Syntax, Keywords
Parameters

Chapter 3

These parameters are used by NGT Reorg and can be overridden in the RRGPARMS DD in the NGT job stream. If not overridden, the default values will be used from the specific system configuration. For more about the system configuration see the BMC Next Generation Technology Installation Guide.

3.1 NGT Reorg Control Datasets

To preserve compatibility with DB2's reorganization utility, NGT Reorg gets its input from four different datasets.

SYSIN (Required)
This dataset contains control statements naming the table spaces, indexes and partitions to be reorganized. See Chapter 2 for a description of the control statements used to run NGT Reorg.

UTLPARMS (Optional)
Contains parameters that control NGT Utilities' behavior.

RRGPARMS (Optional)
Contains parameters that control NGT Reorg's overall behavior.

RRGPARMX (Optional)
Contains parameters that control NGT Reorg index reorganizations.

3.2 Parameter Overview

Parameters coded in the global parameters DD or in any of the utility parameter DDs follow a few simple syntax rules.

1. All parameters must start with a plus sign (+).
2. Parameters can start anywhere, but must be contained within the first 72 character positions of the line. Columns 73-80 are ignored and may contain sequence numbers.
3. Parameters may be coded on multiple lines. Continuation is automatic.
4. Multiple parameters may be coded on the same line.
5. Specify comments by embedding them between /* */ pairs. Comments can be anywhere, even in the middle of a word. All comment strings are removed before syntax checking begins. If specifying parameters via a DD * card in your JCL, be careful not to begin a comment start (/*) in column 1. If you do so, MVS will internally generate a //SYSIN for any subsequent records in the dataset unless you use DLM= to change the JCL comment delimiter.
6. Comments may be nested. There is no (practical) limit to the nesting level of the comments, however, for each /*, there must be a corresponding */.
7. You may specify no value for a parameter by just coding open and close parenthesis with no value, such as +DISP(). This will cause the default value to be used.
8. Blanks, literally coded anywhere, have no significance. All blanks are removed before syntax checking begins. This means that a series of values must be separated by commas.
9. If the same parameter is specified multiple times within the same input dataset, the latest specification prevails.
10. Do not use the plus sign inside the parenthesis for any global or product parameter.

This table lists all the REORG TABLESPACE (RRGPARMS) utility parameters in alphabetical order.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default Value</th>
<th>Use and Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ALLO</td>
<td>No default</td>
<td>Controls value used for primary allocation of new TS or IX</td>
</tr>
<tr>
<td>+BEFOREACC(UT)</td>
<td></td>
<td>Specifies to check for running IBM utilities.</td>
</tr>
<tr>
<td>+FREEPAGE</td>
<td>-1</td>
<td>Number of pages between empty pages in the TS or IX</td>
</tr>
<tr>
<td>+INITWINDOW</td>
<td>Varies</td>
<td>Initial dataspace size</td>
</tr>
<tr>
<td>+IGNDEFNO</td>
<td>NO</td>
<td>Directs processing when deferred indexes exist</td>
</tr>
<tr>
<td>+LOB</td>
<td>NO</td>
<td>Whether to process LOB TS with the base TS.</td>
</tr>
<tr>
<td>+LOBPCT</td>
<td>20</td>
<td>Set page save percent threshold</td>
</tr>
<tr>
<td>+MAXTASKS</td>
<td>10</td>
<td>Maximum number of concurrently executing tasks per Server</td>
</tr>
<tr>
<td>+MINTSSIZE</td>
<td>0</td>
<td>Skip processing for any objects less than the desired size. 0 will process every object</td>
</tr>
<tr>
<td>+PAGEUSE</td>
<td>50</td>
<td>Page Usage Percentage Control</td>
</tr>
<tr>
<td>+PASSWORD</td>
<td>No default</td>
<td>Override password for NGT Reorg</td>
</tr>
<tr>
<td>+PBGCHKDICT</td>
<td>YES</td>
<td>Optionally check the Compression Dictionaries</td>
</tr>
<tr>
<td>+PCTFREE</td>
<td>-1</td>
<td>Percentage of free space in each page</td>
</tr>
<tr>
<td>+RUNSTATS</td>
<td>No default</td>
<td>Activates gathering of stats concurrent with Reorg</td>
</tr>
<tr>
<td>+VOLUMES</td>
<td>No default</td>
<td>List of candidate volumes used for new VSAM dataset</td>
</tr>
</tbody>
</table>

This table lists all the Reorg Index (RRGPARMX) utility parameters in alphabetical order.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default Value</th>
<th>Use and Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ALLO</td>
<td>No default</td>
<td>Controls value used for primary allocation of new TS or IX</td>
</tr>
<tr>
<td>+FREEPAGE</td>
<td>-1</td>
<td>Number of pages between empty pages in the TS or IX</td>
</tr>
<tr>
<td>+PCTFREE</td>
<td>-1</td>
<td>Percentage of free space in each page</td>
</tr>
<tr>
<td>+VOLUMES</td>
<td>No default</td>
<td>List of candidate volumes used for new VSAM dataset</td>
</tr>
</tbody>
</table>
3.2.1  

**+ALLO**  

New Dataset PRIQTY Value

**Syntax**

```
+ALLO
```

**Description**

`+ALLO` in RRGPARMS controls the allocation for the new table space.  
`+ALLO` in RRGPARMX controls the index datasets allocation.

**Operands**

- **ALLO**  
  Specifies that the currently allocated space amount from the existing dataset is to be used as the primary quantity for the new dataset.  The value is obtained from the High-Allocated RBA in the VSAM Catalog.

- **DB2**  
  Specifies that the primary quantity and secondary quantities from the DB2 Catalog are to be used as the values for the new dataset.  If the object is not DB2-managed or if the PQTY value is less than the High-Used RBA for the dataset, then the High-Allocated value is used.

- **KEEPDB2**  
  Specifies that the values from the DB2 Catalog of primary quantity (PQTY) is to be used for the new dataset.  This is different from `+ALLO(DB2)` in that `+ALLO(DB2)` is a conditional switch whereas the KEEPDB2 parameter forces NGT Reorg to use the DB2 values.

  If the dataset is user-managed, the values in use for primary quantity as it is defined to AMS will be used for the new dataset.

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

- **USED**  
  Specifies that the currently used space amount from the existing dataset is to be used as the primary quantity for the new dataset.  The value is obtained from the High-Used RBA in the VSAM Catalog.

- **MXIG**  
  This parameter has a similar meaning to the MVS JCL SPACE parameter.  MXIG requests that space allocated to the data set be the smaller of the PQTY needed, or the space available on the volume.  This parameter affects only primary space allocation.  Secondary extents should be available when using this parameter.

**Example**

```
+ALLO(ALLO)
```

Use the currently allocated space to determine the primary allocation for the new dataset.
3.2.1  +BEFOREACC(UT)
Check for concurrently running IBM Reorg utility and fail if found

Syntax

+BEFOREACC(UT)

Description

The presence of this parameter triggers the product to check the status of the object being reorganized at the beginning and again just before change-apply/rename to check if an IBM Reorg was started during the NGT Reorg. The NGT Reorg is aborted if UT status is found.

If the object is found to be in any UT status the object will be skipped and a message about the restricted state will be issued. If UTRW or UTUT a RC=4 will be produced and if UTRO a RC=8 will be produced.

This check is unnecessary if objects are processed by one vendors utilities or if schedules prevent overlap.

Operands

none  Either specify +BEFOREACC(UT) or don’t.

Example

+BEFOREACC(UT)

Check for UT to find if an IBM utility is running or has started during the NGT Reorg.
3.2.2  **+FREEPAGE**

*New FREEPAGE Value*

**Syntax**

```
+FREEPAGE(n -1)
```

**Description**

`+FREEPAGE` specifies the number of pages between empty pages in the table space. The value specified must be between 0 and 255.

The default value is specified at installation time.

**NOTE**

The DB2 catalog is **NOT** updated with the new value. This will have no effect on DB2 or the table space unless you run the IBM RECOVER, LOAD, or REORG utilities on the object. In that case, DB2 will revert to the original value.

**Operands**

- `-1` Continue to use the FREEPAGE value currently in use.
- `n` An integer representing the new FREEPAGE value.

**Example**

```
+FREEPAGE(35)
```

Insert one empty page after each 35 pages of the table space.
3.2.3  +INITWINDOW
Initial Dataspace Size Control

Syntax

```
+INITWINDOW (n)
```

Description

+INITWINDOW is used by NGT Reorg to limit its memory usage. It is the maximum amount of memory that each task of the server can use. A message will be issued if the object being reorged is too big to run in +INITWINDOW.

The minimum valid value for this parameter is 64; the maximum valid value is 4000. All values are expressed in Megabytes (M).

Operands

- `n` The size of the dataspace in Megabytes.
- `4000` Indicates there is no limit on the amount of memory to be used.

Example

```
+INITWINDOW(2000)
```

Each task of the Reorg server will be allowed to use 2G of memory.
3.2.4  **+IGNDEFNO**  
Whether to ignore DEFINE NO indexes and process the underlying table space

**Syntax**

```
+IGNDEFNO(YES) NO
```

**Description**

+IGNDEFNO is used for multi-table segmented table spaces where some tables are empty and have DEFINE NO or DEFER YES indexes. With +IGNDEFNO(YES) Reorg will process the table space. With +IGNDEFNO(NO) Reorg will produce a warning message and skip the object.

+IGNDEFNO(YES) is popular with SAP users who have many empty table spaces with deferred indexes.

**Operands**

- **YES**  
  Ignore deferred indexes on empty tables and process the table space for the other tables in the table space.

- **NO**  
  Do not process table spaces that have deferred indexes.

**Example**

```
+IGNDEFNO(YES)
```

Process table spaces that have deferred indexes on empty tables.
3.2.5  +LOB
Whether to include LOB auxiliary table spaces when processing a table space list

Syntax

Syntax

+LOB(
  NO
  YES
)

Description

+LOB(NO), the default, is used to exclude LOB auxiliary table spaces when processing a wildcard list of table spaces. +LOB(YES) is used when you want to include LOB auxiliary table spaces in a wildcard list of table spaces.

☑️  NOTE:
The AUX keyword introduced with NGT LOBMaster overrides this +LOB parm. If +LOB(NO) and AUX are specified, The LOB TS will be processed. LOBMaster is a separate product with a separate manual.

Operands

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Default. Do not Reorg associated LOB auxiliary table spaces.</td>
</tr>
<tr>
<td>YES</td>
<td>Process associated LOB table spaces when reorging a base table space.</td>
</tr>
</tbody>
</table>

Example

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>+LOB(YES)</td>
</tr>
</tbody>
</table>

Process all LOB table spaces when Reorging the base table space.
3.2.6 +LOBPCT

LOB space percent saved parm.

Syntax

```
+LOBPCT(pctsave)
```

Description

LOBPCT is used to set a LOB percent space saving threshold to prevent unnecessary costly reorganizations. This parameter can be overridden by the NGT Reorg keyword LOBRECLAIMPERCENT. If the Reorg will not reclaim the specified percent the object will be excluded.

Operands

- **pctsave**
  - Default 20. Process LOB auxiliary table space only if pctsave space can be saved. Set to zero to force the Reorg regardless of space to be saved.

Example

```
+LOBPCT(99)
```

Process the object when triggered regardless of space savings.
3.2.7 **+MAXTASKS**
Server Task and Parallel Processing Control

**Syntax**

```
+MAXTASKS(n)
```

**Description**

`+MAXTASKS` specifies the maximum number of concurrently executing tasks **per server**. The value specified must be between 1 and 15.

**Operands**

- **10**
  Default. Processing will run up to 10 tasks simultaneously, per server.
- **n**
  An integer between 1 and 15 representing the maximum number of tasks per server.

**Usage Notes**

`+MAXTASKS` can have an "absolute" maximum of 15 tasks per server. If a lower maximum number of server tasks is desired — say, 7 — then no more than 7 tasks will be spawned **per server**. So if `+SVRDISK(3)` and `+MAXTASKS(7)` are specified together, then no more than 21 server tasks will ever be running at once, even though with an absolute maximum of 15 (that is `+MAXTASKS(15)`, the highest valid value possible), up to 45 tasks could be running.

**Example**

```
+MAXTASKS(5)
```

Restrict parallel processing to five concurrent server tasks.
3.2.8  **+MINTSSIZE**

Minimum table space Size Control

**Syntax**

```
+MINTSSIZE(0 number)
```

**Description**

+MINTSSIZE specifies the minimum table space size, in pages, for which a Reorg will be performed. If specified, a Reorg will be bypassed for any table space that is smaller than the specified value.

Valid values are either 0, or a number ranging from 1024 up to 2147483648 (or 2G pages). Values are expressed in pages, without the use of "K", "M" or "G" units.

If a value of 0 is specified, the processing will automatically reorg all objects – just as processing would occur without this parameter's processing in place.

**Operands**

- **0**  
  Minimum table space size check is bypassed; all table spaces are processed.

- **number**  
  A numeric value from 1024 pages up to 2147483648 (2G) pages.

**Examples**

```
+MINTSSIZE(65536)
```

Processing will bypass a Reorg of table spaces smaller than 65536 pages (or 64K pages) in size. All table spaces of at least 64K pages in size will be considered for processing.

```
+MINTSSIZE(1073741824)
```

Processing will bypass a Reorg of table spaces smaller than 1G pages in size.

```
+MINTSSIZE(0)
```

Processing will Reorg all table spaces specified in the SYSIN, without checking their sizes beforehand.
3.2.9  +PAGEUSE
Page Usage Percentage Control

Syntax

\[ +\text{PAGEUSE}(n) \]

Description

Specifies the percentage of available memory to be used.

Operands

\( n \)  
An integer from 1 to 99.

Examples

\[ +\text{PAGEUSE}(75) \]

Use 75 percent of available pages, leaving 25 percent blank.

NOTE

+PAGEUSE(0) will default to (50). +PAGEUSE(50) is the default and is the recommended initial value.
3.2.10  **+PASSWORD**  
License Expiration Password for NGT Reorg

**Syntax**

```
+PASSWORD(string)
```

**Description**

Obtain the password string value from BMC Customer Support. The password may be supplied to extend the no-obligation trial or to set the utility as non-expiring after license. +PASSWORD is usually specified at installation time, eliminating the need to specify it in each job.

**Operands**

- **string**  
  A character string representing the password for NGT Reorg.
3.2.11  **+PBGCHKDICT**

Optionally perform the PBG Compression Dictionary check.

**Syntax**

```
+PBGCHKDICT( NO )
```

**Description**

+PBGCHKDICT(NO) specifies to skip the compression dictionary check and assume all partitions have the same compression dictionary as they should. If (YES) the time to check is saved and an abend will occur if they are later found to not be the same.

+PBGCHKDICT(YES), the default, specifies to check the compression dictionary and make them all the same if they are not.

**Operands**

- **NO**  
  Skip the compression dictionary check.
- **YES**  
  Check compression dictionaries.

**Example**

```
+PBGCHKDICT(NO)
```

Skip the dictionary check; it is known that they are all the same.
### 3.2.12 +PCTFREE

**New PCTFREE Value**

**Syntax**

```
+PCTFREE(n)
```

**Description**

+PCTFREE specifies the percentage of free space in each data page of the table space. The value specified must be between 0 and 99.

The default value is specified at installation time.

**NOTE**

The DB2 catalog is NOT updated with the new value. This will have no effect on DB2 or the table space unless you run the IBM RECOVER, LOAD, or REORG utilities on the object. In that case, DB2 will revert to the original value.

**Operands**

- `-1` Continue to use the PCTFREE value currently in use.
- `n` An integer representing the new PCTFREE value.

**Example**

```
+PCTFREE(-1)
```

Use the PCTFREE value from the DB2 catalog for each object.
3.2.13 **RUNSTATS**

Updates Catalog Statistics

**Syntax**

```
+RUNSTATS( NO YES )
```

**Description**

Activates statistics gathering and updates the DB2 catalog.

**Operands**

- **NO**  Do not update the statistics.
- **YES**  Update the statistics, according to the STTPARMS\(^1\).
- **ALL**  Same as YES. Update the statistics, according to the STTPARMS.

\(^1\) STTPARMS DD is where NGT Stats parameters are specified. See the *BMC Next Generation Technology Stats for DB2 for z/OS Reference Manual* for more details.

**Restrictions**

The following restrictions apply to the use of this parameter.

- This parameter will compute aggregate statistics using catalog values for partitions that are not processed. In order to ensure that meaningful values are computed it is necessary to have collected statistics for all partitions before processing a subset of a partitioned table space.
- Non-indexed tables will be skipped.
- The stats gathered with Reorg will only update SYSCOLUMNS for the first column of each index. Run NGT Runstats to get stats for all columns of all indexes.

**Example**

```
+RUNSTATS(YES)
```

Gather statistics and update the DB2 Catalog with the new values.

Note: NGT Utilities always gather KEYCARD stats when any stats are requested.

**NOTE**

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

Column Grouping (COLGROUP) statistics will be updated for every column combination of every index. Frequency distributed statistics can also be collected. Frequency stats collection is controlled by the +PARMS for the NGT Stats product using the STTPARMS DD. If the STTPARMS DD is not supplying these +parms in the Reorg job, they will be obtained from the NGT Configuration.

For a list of what DB2 Catalog statistics are updated, see the NGT Stats manual.

Example

To gather column Frequency distributed stats with Reorg.

RRGPARMS DD *
  +RUNSTATS(YES)
STTPARMS DD *
  +FREQ(MOST)
  +NUMCOLS(4)
  +COUNT(2)

Note: If one of the indexes has only three columns, frequency stats will be gathered for that group of three columns.
3.2.14 +VOLUMES
Volume Serials for new IDCAMS DEFINE Operations

Syntax

\[ +\text{VOLUMES} (\text{volserlist}) \]

Description

+VOLUMES specifies a list of volumes to be used in the DEFINE for each new dataset. Up to eight volume serials (volsers) can be specified, each separated from the next by a comma; any volsers in excess of eight will be ignored. The volume serials list can be continued onto multiple lines. A default list may be specified at installation time; it can be overridden by specifying +VOLUMES( ) at execution time. If no volume override list is specified, the new dataset is DEFINEd on the same list of volumes that the original dataset used, obtained from a LISTCAT of the original dataset.

It is not necessarily the case that +VOLUMES represents a list of volume serials. It only represents a value that is substituted into a model AMS DEFINE statement. This model DEFINE statement can be modified for your installation, for example, to support the use of SMS. If the variable &VOLUMES is used in a DEFINE keyword other than VOLUMES(&VOLUMES), like MANAGEMENTCLASS(&VOLUMES), the value specified on the +VOLUMES parameter would be substituted for it.

NOTE:
The single * is for single volume requests and the *,* is for multi-volume requests. Use +VOL(*,*) for three volumes. If however, you prefer to allocate volumes as they are needed, use +VOL(*) with a single * and use VOLUMES="*" in XSUTXTND to add volumes when extend is necessary. Note: XSUTALLO overrides this parameter.

Operands

\text{volserlist} \quad \text{A list of volume serials (volsers), with names separated by commas.}

Example

\[ +\text{VOLUMES}(\*) \]

Substitute "*" for the &VOLUMES variable in the DEFINE statement.

Usage Notes

\textbf{Conditional Volumes.} A special form of the +VOLUMES parameter specifies that a subset of the volumes are to be chosen from the list based on the space available on the volume just before allocation is to occur. This is the conditional form of the parameter. When the first value specified in the +VOLUMES parameter is "/n" (n is a number from 1 to 8), then "n" volumes will be chosen from the list based on the space available on the volume. "n" is the number of volumes to include in the final list.

For example:
requests a check of the space available on the four volumes specified to find one volume that contains contiguous space for the PRIQTY value of the dataset. The volume with the largest contiguous free space will be used. Only the one volume will be used for the dataset because "/1" was specified. To include multiple volumes in the final allocation list, specify "/n", where "n" is more than 1. Additional volumes will be specified as candidate volumes starting at the beginning of the list, not including the one picked for the primary volume, until the count is reached. For example, if you specified

+VOLUMES(/4,AAAA,BBBB,CCCC,DDDD,EEEE)

and volume CCCC has the largest contiguous free space of any of the named five volumes, then the volumes list passed for dataset allocation would be

(CCCC,AAAA,BBBB,DDDD)
Chapter 4  **Examples**  This chapter covers the syntax for specific examples of NGT Reorg.

4.1.1  **Reorganize a table space**
To reorganize a table space, use the keywords REORG TABLESPACE.

```plaintext
REORG TABLESPACE NWDB.ALLACCTS
```

This syntax works for all types of table spaces – simple, segmented and partitioned.

4.1.2  **Reorganize specific parts of a partitioned table space**
To reorganize specific (physical) parts of a partitioned table space, use the PART or PARTS keyword. If you need to reorganize only one part of a table space, you use PART:

```plaintext
REORG TABLESPACE NWDB.ALLACCTS PART(7)
```

While if you need to reorganize multiple parts, you specify the starting and ending partition numbers separated by a colon, following the PARTS keyword:

```plaintext
REORG TABLESPACE NWDB.ALLACCTS PARTS(2:8)
```

For further flexibility single partitions and partition ranges can be specified together as follows:

```plaintext
REORG TABLESPACE NWDB.ALLACCTS PARTS(2,5:9,12,15:22,25,30)
```

4.1.3  **Reorganize specific logical parts of a partitioned table space**
To reorganize specific logical parts of a partitioned table space (designed by use of DB2’s ROTATE feature), use the LPART or LPARTS keyword. If you need to reorganize only one part of a table space, you use LPART:

```plaintext
REORG TABLESPACE NWDB.ALLACCTS LPART(7)
```

While if you need to reorganize multiple parts, you specify the starting and ending partition numbers separated by a colon, in parentheses, following the LPARTS keyword:

```plaintext
REORG TABLESPACE NWDB.ALLACCTS LPARTS(2:8)
```

For further flexibility single partitions and partition ranges can be specified together as follows:

```plaintext
REORG TABLESPACE NWDB.ALLACCTS LPARTS(2,5:9,12,15:22,25,30)
```
4.1.4 **Reorganize a table space and generate an image copy**

It is a regular feature of NGT Reorg to automatically produce an image copy after each REORG, so no statement is necessary to that effect.

**NOTE**

You may control the naming of image copy datasets with the automation control point `XCPYDYNM`. For details on the use of `XCPYDYNM`, please see the *BMC Next Generation Technology Automation Reference Guide*.

4.1.5 **Reorganize a table space, sending discards to a dataset**

To reorganize a tablespace and send discards to a dataset based on a specific criteria, use the DISCARD FROM TABLE clause and include a WHERE criteria:

```sql
//OBSOLETE DD DISP=SHR,DSN=ADMIN.CENTRAL.DISCARDS
//SYSIN DD *
REORG TABLESPACE DB9900.CENTRAL
  DISCARDDN(OBSOLETE)
  DISCARD FROM TABLE DB9900.CENTRAL_INV_TBL WHERE (SKU > '592032')
```

4.1.6 **Reorganize a set of table spaces, excluding certain table spaces**

You can reorganize a set of table spaces using wildcards and then exclude those you don't wish to include with the EXCLUDE TABLESPACE keywords.

```sql
REORG TABLESPACE ADMINDB.CAR%
  EXCLUDE TABLESPACE ADMINDB.CARMIDSZ
  EXCLUDE TABLESPACE ADMINDB.CARLUXRY
  EXCLUDE TABLESPACE ADMINDB.CARSUVS
```

4.2 **NGT Reorg Index**

4.2.1 **Reorganize an index**

To reorganize an index, use the keywords REORG INDEX.

```sql
REORG INDEX NWDB.XACCT912
```

This syntax works for indexes on all types of table spaces – simple, segmented and partitioned.

4.2.2 **Reorganize specific parts of an index**

To reorganize specific parts of a partitioned index, use the PART or PARTS keyword. If only one part of the index needs to be reorganized, the keyword PART is used:

```sql
REORG INDEX NWDB.ARIX1 PART 7
```

While if a reorg of multiple parts is needed, the starting and ending partition numbers are specified following the PARTS keyword, separated by a colon:
Examples

REORG INDEX NWDB.ARIX1 PARTS 2:8

For further flexibility single partitions and partition ranges can be specified together as follows:

REORG INDEX NWDB.ARIX1 LPARTS(2,5:9,12,15:22,25,30)

4.2.3 Reorganize a set of indexes excluding certain indexes

Similar to REORG TABLESPACE, you can reorganize a set of indexes using wildcards and then exclude those you don't wish to include with the EXCLUDE INDEX keywords.

REORG INDEXES ADMINDB.CAR%
  EXCLUDE INDEX ADMINDB.XMIDSZ1
  EXCLUDE INDEX ADMINDB.XLUXRY2
  EXCLUDE INDEX ADMINDB.XSUVS8

For further flexibility single partitions and partition ranges can be specified together as follows:

REORG TABLESPACE NWDB.ALLACCTS LPARTS(2,5:9,12,15:22,25,30)
Summary Reports

Chapter 5

NGT Reorg produces a series of reports to the SUMMARY DD.

There are other reports such as the RTS REORG reports if RTMAN is used. And the OBJECTS—COPIED report. These other reports are documented in their corresponding manuals.

5.1 Internal IDS Report

The Internal IDS Report is a brief report provided to correlate the DB2 internal object identifiers to their full external names. This can be useful when interpreting some product messages where the internal identifier is used.

<table>
<thead>
<tr>
<th>TABLESPACE</th>
<th>DB-0110</th>
<th>PS-0002</th>
<th>FILE-0001</th>
<th>PTNS-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBL 0003</td>
<td>TEST_0040_TBL0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TPLIX TESTDBAS.FAKE PIX</td>
<td>FANSET=0000</td>
<td>PGSET=0000</td>
<td>TYPE=DF</td>
<td>KYLEN=13</td>
</tr>
<tr>
<td>C PIX TESTDBAS.TESTIX01</td>
<td>FANSET=0004</td>
<td>PGSET=0005</td>
<td>TYPE=DF</td>
<td>KYLEN=13</td>
</tr>
<tr>
<td>NPI TESTDBAS.TESTIX02</td>
<td>FANSET=0006</td>
<td>PGSET=0007</td>
<td>TYPE=DF</td>
<td>KYLEN=24</td>
</tr>
<tr>
<td>DPSI TESTDBAS.TESTIX03</td>
<td>FANSET=0008</td>
<td>PGSET=0009</td>
<td>TYPE=DF</td>
<td>KYLEN=10</td>
</tr>
</tbody>
</table>

Figure 5-1. Internal IDS Report

Each line of this report is part of the object being processed.

The first line is the TABLESPACE where it names the database.tablespace followed by the HEX value for the DB (DBID), PS (PSID), FILE (OBID), and the number of partitions.

The second line is the TBL showing its OBID and name.

The third line in this case is a fake partitioning index for internal use. It is used on table-partitioned table spaces.

The remaining lines show the various types of indexes and their names; followed by their:

- **FANSET**: OBID
- **PGSET**: ISOBID
- **TYPE**: First character: Dup or Unique, second character: Fixed or Variable.
- **KYLEN**: Key length in bytes
- **KYCOLS**: Number of table columns in the index
5.2 **Object Space Report**

The Object Space Report documents the type and size of the objects being processed.

![Object Space Report](image1.png)

Figure 5-2. Object Space Report

5.3 **Before-After Report (BF AF)**

The Before-After report documents rows, pages, and index metrics prior to and after the reorg.

![Before-After Report](image2.png)

Figure 5-3. Before-After Report
Chapter 6  

Relevant Automation

NGT Automation control points are integrated into all NGT Utilities and can greatly enhance and expand processing options. This chapter presents automation control points recommended but not required.

The presence of the CDBEXEC DD statement triggers automation control point processing, as in this example:

```
//CDBEXEC DD DISP=SHR,DSN=NGT.CDBEXEC
```

The automation control point dataset will contain one member with each of the control points chosen for use.

In listing the automation control points below, be aware that these are not the only ones that can be used with this product. These should serve only as a basic recommendation for automation. This recommendation should be used to spawn more ideas or give insight into how others are utilizing this powerful facility. Any of the automation control points may be incorporated for whatever uses are required.

The following standard NGT automation control points may be used with this utility:

6.1 **XRRGINIT (Reorg Initialization)**

Invoked to control REORG. For the REORG of a table space, it is invoked after the Clustering Index has been read and current statistics have been obtained. For the REORG of an index, it is invoked before the index leaf pages are read. This automation control point can bypass the operation completely based on current clustering index statistics or for any other reason appropriate to your installation.

The decision to Reorg can now be made with the BMC Next Generation Technology Utility Manager (NGT Utility Manager). Using the NGT Utility Manager is much more efficient.

6.2 **XRRGTERM (REORG Completion)**

Invoked as REORG completes for each object being processed. It can save appropriate statistics, either in a file or in a DB2 table.

6.3 **XRRGRWRO**

Invoked just before the Drain in preparation for the Change Apply phase. The automation control point can wait for a specific time or wait on some external event to manipulate when the Change Apply phase starts. This automation control point is called by the master job only, not by the servers.

This automation routine can be used to affect a delay between drain attempts during Change Apply. Just copy member XRRGRWRO from CNTL to your CDBEXEC and change "delay = 0" to the number of seconds delay you want.
6.4 **General Utility Automation Control Points**

**XSUTGLOB**  Can be used by server jobs for early access to global variables. Suitable place for declaring "global-global" variables (variables "visible" across all server jobs and their associated automation control points that will be activated).

**XSUT0000**  Can be used by the master job for early setting of global and global-global variables. Run by master jobs only, not called by server jobs.

**XSUTSYIN**  Called before the parser reads SYSIN. It has full access to the SYSIN and can modify it. It may be used to override or disallow some input parameters. Run by master jobs only, not called by server jobs.

**XSUTTIME**  Called to allow canceling utility based on time of day. Run by master jobs only, not called by server jobs.

**XSUTDBMG**  Called to set volumes for initial allocation and extend processing.

**XSUTTERM**  Called after processing has finished. It may be used, for example, to insert run statistics into a DB2 table or to e-mail an audit report at the completion of processing, automatically.

**XSVRERR**  Called if processing ends with an error condition. It may be used to alert a user, raise an error flag, or perform cleanup.

For further information and details on the use of all of these automation control points, please see the *BMC Next Generation Technology Automation Reference Guide*. 
Chapter 7  Technical Details

7.1  Reasons to Reorg

What should be reorganized, indexes or table spaces? In general, table spaces are over reorganized, and indexes are under reorganized. There are five basic reasons to reorganize a table space:

Reclaim space from dropped tables.

Eliminate pointer-overflow pairs. When a row increases in size, it may no longer fit in the space reserved for it in the page. When this happens, DB2 moves the row to a new location and places in the original spot a pointer to this overflow location. Note that this happens only if your rows contain variable columns (e.g., VARCHAR) that significantly increase in size as the result of SQL UPDATE statements, or if compression of the rows becomes less efficient after an update.

Reestablish free space. Reestablishing free space may be counter-productive. If an application has heavy inserts and inadequate free space, it may seem at first glance that reorganizing the table space may improve performance by reestablishing free space for subsequent inserts. Often the opposite is the case. Table space scans take longer because there are more pages to scan. In either case, the benefit or penalty is small. If the table space is ordered by a clustering index, then the amount of free space assumes more significance. Availability of free space can allow DB2 to maintain clustering. Conversely, a free space shortage can cause inserted rows to be placed out of clustering order.

Collecting segments in segmented table spaces. This is relevant only to segmented table spaces. Adjacency of segments is important only if the table space is being scanned. Avoiding table space scans is a much better idea than trying to make them more effective. reorganizing an index may make it more attractive to DB2, thus providing a much more significant performance advantage.

Reestablishing clustering. This provides a real performance advantage. However, the advantage is by no means universal. In order to be meaningful:

- A clustering index must exist,
- DB2 must be able to use the index for the query at hand,
- DB2 chooses not to use the index because the clustering ratio is too low, and
- DB2 would use the index if the clustering ratio were higher.
7.1.1 Reasons to Not Reorg
There are also reasons NOT to reorganize a table space.

Some access paths are NEVER helped by a table space reorganization. For example, index scans for non-clustering indexes and direct access to the table space via the index.

Table space Reorg primarily improves the efficiency of table space scans (and often not significantly). It is a much better idea to avoid table space scans, especially with large tables.

Table space Reorgs are more expensive in terms of system resources used (CPU and DASD) than Index reorgs. Table space Reorgs reorganize every index in the table space, even indexes that do not need reorganizing.

7.1.2 Index Reorg Advantages
Index reorgs provide multiple advantages.

Index reorgs improve the performance of your critical queries. After an index reorg, queries take less time to complete and use less system resources, improving the overall performance of applications.

An index reorg may reduce the number of levels in the index. This improves the speed of index searches.

Index reorgs may reduce the number of leaf pages, improving the speed of index scans.

Index reorgs restore a reasonable free space specification. Over time, many indexes tend to increase the amount of free space they hold, since page splits produce pages that are only 50% full.

Reorganizing an index may reduce the number of extents in the index dataset. This improves the physical access performance to the index.

Index disorganization has a critical effect on overall application performance. More queries are affected more severely by a disorganized index than by a disorganized table space.

Finally, indexes are more volatile and disorganize faster than table spaces. For example, as pages split, freespace tends rapidly to 50%; as a result, the number of levels grows unnecessarily; also the distance between adjacent pages increases, a factor that is unimportant in most table spaces but critical in index access paths. Indexes should be reorganized as often as possible. Index reorg consumes only a fraction of the resources required to complete a full table space reorg.

7.1.3 Reorganizing Partition By Growth (PBG) table spaces
PBG table spaces are capable of being enormous. The resources required to reorganize a PBG table space can be significant. NGT Reorg currently does not support processing select partitions of a PBG. Reorganizing one part is of limited use since rows must stay in that part when it really belongs in another part. Reorganizing one part of a PBG would be the equivalent to reorganizing one dataset of a classic segmented table space.

The NOCLIX keyword is offered to have a Partition By Growth table space reorged as if there were no Clustering IX. This will reclaim space only.

The MAXDISORG n keyword is offered to cancel the reorg if there is too high of a percentage of disorganization. This can be used to prevent unexpected long running reorgs on very large table spaces. If such a PBG is encountered NGT Reorg will report that it was skipped. Plans
should be made to Reorg such table spaces with MAXDISORG set very high when the extended elapsed time is more acceptable. The default value for MAXDISORG is 50%, no table space should be allowed to get that disorganized.

After a NGT Reorg all partitions of the PBG table space will have the same compression dictionary. A new compression dictionary will be produced and compared to the existing part 1 dictionary; the better of the two will be used for all parts. This means NGT Reorg may have to decompress and compress each row.

7.2 Fine Tuning for the NGT REORG Utility
The defaults shipped with NGT Reorg are appropriate for the majority of environments. It is possible, however, to override these defaults in order to address specific needs. Defaults are changed via a set of parameters that can be passed to NGT Reorg at execution time. See "Reorg Parameters Summary".

7.2.1 Automatic Restart
NGT Reorg avoids pitfalls by treating all invocations as restarts, and reversing the typical treatment of normal and exceptional executions.

Normal executions of NGT Reorg, i.e., those which have either never been run before, or which ran successfully to completion the immediately prior time, are treated as being exceptional in that no restart processing is necessary. Execution therefore begins at the beginning.

Exceptional executions of NGT Reorg, i.e., those which failed in some way the immediately prior time they were run, are treated as being normal in that the automatic restart processing is required.

Unlike with most programs, if an execution of NGT Reorg fails, all that is necessary for the execution to be restarted is that the job be resubmitted. No changes are needed, period. NGT Reorg will determine that the prior execution terminated prematurely, and will resume execution as is appropriate.

7.2.2 NGT Reorg vs. IBM Reorg Restart
NGT Reorg Version 9 uses an internal matrix of utility IDs and objects that are being processed. This is to prevent two incompatible utilities from executing on the same object concurrently. Like IBM it is now often necessary to deal with a previously failed utility ID prior to running a utility with a different utility ID on the same object. Just as before with NGT utilities, the previous utility can be run with a RESTART or with QUICKEXIT to just clean up any temporary files. If the utility is in a "must complete" state and a QUICKEXIT is run the utility will simply complete. NGT Reorg ensures no damage can occur by resubmitting. NGT Version 9 Auto-Online Reorg always runs with the object in RW status and leaves the original table space and indexes untouched until the Reorg is complete. For more information, see the \textit{BMC Next Generation Technology General User Guide}.

7.2.3 Specifying a Utility ID
The key to this process is the utility ID. NGT Reorg accepts a 16-character utility ID as part of the PARM field on the JCL EXEC statement. Any value 16 bytes or less in length is acceptable to NGT Reorg.

If NGT Reorg determines that a prior execution using this same utility ID ended prematurely, then this execution proceeds as follows.
Technical Details

- If no restart option is specified then execution resumes from the most recent successful checkpoint, which is normally the last completed table space or partition.
- If a restart option is specified then
  - if your choice is feasible in the actual circumstances, execution proceeds as dictated.
  - if the restart option is incompatible with the actual circumstances, then execution stops.

Remember, the key to this process is the utility ID. Points to consider include the following.

1. It is the reorganization of the original set of objects that is resumed. If the control statements are changed (i.e., those that specify the objects to be reorganized) before resubmitting the utility, these new statements are ignored.
2. Two NGT Reorg steps using the same utility-id in different jobs cannot run concurrently. This is the case whether the two steps appear in inadvertent duplicates of the same job or in unrelated jobs which happen to be using the same utility-id. The second step to begin execution will fail.

7.2.4 The Checkpoint Dataset

Every execution of NGT Reorg requires access to a linear VSAM dataset named during the installation procedure and configured with the //CDBCKPT DD statement. This Checkpoint dataset (CDBCKPT) contains all the restart information for utilities.

The checkpoint dataset has information such as:

1. The list of objects to be Reorged and the status of each.
2. The required internal tables to be able to restart.
3. The list of objects being renamed.
4. The RUNSTATS values.

For further information on the checkpoint dataset, see the BMC Next Generation Technology General User Guide.

7.3 Specifying Free Space

NGT Reorg provides free space specification options within the Reorg utility.

Two free space specification options are available for table spaces under RRGPARMS DD and the same two are available under RRXPARMS DD for indexes.

- `+FREEPAGE`  How often to leave empty y pages in the dataset.
- `+PCTFREE`   The percentage of free space in each page.

These parameters have the same meaning and effect as the DB2 parameters with the same names.
7.4 Processing DB2 objects that are Reorg Pending

Because of the enhanced processes used by BMC NGT, NGT Reorg is significantly faster and uses fewer resources than competitive tools. However, these different processes and the various reasons an object could be Reorg Pending had prevented NGT Reorg from processing objects that are unavailable due to being Reorg Pending. NGT Reorg has always processed objects that are Advisory Reorg Pending.

NGT Reorg will process objects that are Reorg Pending to make them available. This will not be the usual high speed NGT Reorg, but a more traditional Reload Reorg. While processing REORP objects Reorg will make use of the Load parameters specified in the LODPARMS DD. This includes common parameters which also exist in RRGPARMS such as +ALLO

![NOTE](https://example.com/note_icon.png)

Some restrictions apply when the partitioned table space that is REORP also has associated LOB table spaces. See Processing LOB table spaces below.

If the TS is REORP due to altering limitkeys, and the limitkey for the last part was altered in such a way that rows will be discarded; NGT Reorg will not process this case. If this case is needed, discard the records prior to the Alter limitkey.

7.4.1 Minimize down-time due to REORP

Since Reorg Pending objects are completely unavailable; this status should be avoided when possible, and preplanned to minimize the down time when necessary. The following are recommendations to minimize this down time.

1. Reorganize the object to be altered into a REORP state.
2. Perform the ALTER that will render the object Reorg Pending and unavailable.
3. Reorg the object with load parameter +SORT(NO) to eliminate the Reorg Pending and make the object RW again.

While this process maximizes the objects availability, it is not necessary. If the down-time is not an issue then just Alter the object and run Reorg with the +SORT(YES) load parameter to clear the Reorg Pending status and organize the data. The regularly scheduled reorg job would process any other parts that meet the criteria for reorganization.

7.4.2 DDL That cause REORP status

Currently these actions will render a table space Reorg Pending making it unavailable to all SQL. Below is just a summary of these actions; we recommend that you always test such a change to know if the DDL you plan to run will render an object Reorg Pending.

- Alter table alter column of partitioning index, increase character length
- Alter Limitkey (except for the last part)
- Alter add partition (If boundary for last part not enforced)
- Rotate (if boundary for last part not enforced)

7.5 Processing DB2 indexes that are Rebuild Pending

The NGT Reorg is significantly faster and uses fewer resources due in part to Reorganizing the table space and its indexes concurrently. Because NGT reorganizes indexes rather than rebuilding them, indexes that are in Rebuild Pending status cannot be processed by NGT Reorg.
7.5.1 Minimize down-time due to RBDP or PSRBD

Since Reorg Pending objects are completely unavailable; this status should be avoided when possible, and preplanned to minimize the down time when necessary. The following are recommendations to minimize this down time.

1. Perform the ALTER that will render the index Rebuild Pending and unavailable.
2. Run REBUILD INDEX to both eliminate the Rebuild Pending making it RW again and reorganize the index.

The regularly scheduled reorg job would process any other index, index part, or table space part which meets the NGT Utility Manager criteria for reorganization.

7.5.2 DDL That cause RBDP or PSRBD status

Currently these actions will render an index Rebuild Pending making it unavailable. Below is just a summary of these actions; we recommend that you always test such a change to know if the DDL you plan to run will render an index Rebuild Pending.

- Alter PIECESIZE
- Alter indexed variable length column between PADDED and NOT PADDED
- Alter add column to TB and IX, and Insert rows all in the same commit scope
- Alter indexed column from SMALLINT, INTEGER, or DECIMAL to BIGINT
- Alter indexed column from CHAR FOR BIT DATA or VARCHAR FOR BIT DATA to BINARY or VARBINARY
- Alter indexed DECIMAL column
Materializing On-Line Schema changes
NGT Reorg will check for pending schema changes and materialize them during the reorganization.

Processing the pending DDL requires NGT Reorg to process all parts and post them simultaneously. This prevents the individual partition posting that occurs with partition independence when there are no NPIs.

If there are pending changes and a Reorg attempts to process a subset of the partitions, the object will be excluded from processing with message: NGTR790 NO PART SELECTION ALLOWED ON A REORG THAT IS PROCESSING PENDING ALTERS
To avoid skipping needed reorgs, it is advisable to issue the Alter just prior to Reorganizing the entire object.

During the Reorg you will see a utility posted in SYSIBM.SYSUTILX. An IBM –DIS UTIL will show something similar to the following: (Do Not Terminate this UTIL)

DSNU100I -DB2P DSNUGDIS - USERID = userid
MEMBER = DB21
UTILID = CDBreorguid.0002
PROCESSING UTILITY STATEMENT 1
UTILITY = RUNSTATS
PHASE = UTILINIT COUNT = 0
NUMBER OF OBJECTS IN LIST = 1
LAST OBJECT STARTED = 1
STATUS = STOPPED

You will see a stopped Runstats utility with a utility ID starting with "CDB" followed by the NGT Reorg utility ID. (Do Not Terminate this UTIL)

The object will show a status of UTRW during the NGT Reorg. This only occurs when there are pending changes. This is done to ensure no other Alters can be run that add records to SYSPENDINGDDL for this object while Reorg is processing current pending DDL. NGT Reorg will process the pending DDL it finds for an object when it starts and clear all records for the object when complete. This stopped Runstats utility prevents further Alters without restricting access to the object during the Reorg. (Do Not Terminate this UTIL)
7.7 Processing table spaces with LOBs

NGT Reorg processes LOB table spaces and their Base table spaces separately. With a NGT LOBMaster license the Base and LOB Auxiliary table spaces are Checked and Reorged together online. For more information about LOBMaster see the BMC Next Generation Technology LOBMaster for DB2 for z/OS Reference Manual.

When you specify a wildcard list of table spaces to process you can omit LOB table spaces by specifying +LOB(NO). +LOB(NO) is the default. With +LOB(YES) the product will process all base and LOB table spaces included by the wildcard specified.

Currently LOB auxiliary table spaces processed without LOBMaster are not processed online; they are put in UT status.

Since Base table spaces with LOB columns and their associated LOB table spaces are processed separately, the following restrictions apply.

Base table space with LOB columns.
When PBG table spaces are reorganized rows are moved between partitions. Since the LOB rows in the associated LOB table spaces must be moved too, and the product processes the LOB table spaces separately, this is not allowed. NGT Reorg currently does not support PBG table spaces with LOB columns.

PBR base table space with LOB columns and ALTER LIMITKEY has been run causing REORP.
The Reorg to remove the REORP status may have to move rows between partitions. Since the LOB rows in the associated LOB table spaces must be moved too, and NGT reorg processes the LOB table spaces separately, NGT Reorg will not process this case. This is true for any ALTER that forces the table space to be unavailable with the REORP status such as certain Alters to Add Partition and Rotate.

Reorg with Discarding from a TS with LOB columns.
Discarding rows from the base table requires discarding the associated rows from the LOB table spaces. Because the NGT product is currently processing the base and aux table spaces separately, any Reorg with discards from the base table will place the LOB auxiliary table in ACHKP. Such a Reorg should be followed with IBM Check utilities to find and delete any orphaned LOB rows. This restriction is lifted by NGT LOBMaster.
Index

/
/1, 3-19

+
+ALLO, 3-3
+FREEPAGE, 3-5, 7-4
+IGNDEFNO, 3-7
+INITWINDOW, 3-6
+LOB, 3-8
+LOBPCT, 3-9
+MAXTASKS, 3-10
+MINTSIZE, 3-11
+PAGEUSE, 3-12
+PASSWORD, 3-13
+PBGCHKDICT, 3-14
+PCTFREE, 3-15, 7-4
+RUNSTATS, 3-16
+VOLUMES, 3-18

A

ACHKP, 7-8
Administrator, 1-3
ALLO, 3-3
AMS, 3-3
AMS DEFINE, 3-18
ASCIIDEL, 2-10
authority, 1-3
Authorizations, 1-1, 1-3
Automatic Restart, 7-3
automation exit processing, 6-1
AUX, 2-6
Auxiliary table space, 1-1

B

BASIC, 2-7
BEFOREACC(UT), 3-4
Before-After Report, 5-2
BRF, 2-6

C

CDBEXEC, 2-2, 6-1
Cloned Tables, 1-1
COMPRESS, 2-6
Concurrent Copy, 1-4
Conditional Volumes, 3-18

COPYDDN, 2-7

D
data integrity, 1-4
data page, 3-15
DB2, 3-3
DB2 States, 1-2, 1-3
DBADM, 1-1, 1-3
dbmask.spacemask, 2-5
dbname.spacename, 2-5
DD Statements, 2-1
DDL, 1-1, 1-3
delay, 6-1
Delete, 1-4
deleted rows, 2-8
DISCARD, 2-6, 2-11
DISCARD FROM TABLE, 2-10
DISCARDDN, 2-8
discards, 4-2
DSNDB04, 2-5
DSNTIAUL, 2-9
DSNUM ALL, 2-6, 7-8
DSSIZE, 1-5

E

EXCLUDE, 2-10, 2-13
EXTENDED, 2-7
EXTERNAL, 2-9

F
Features, 1-4
FLASHCOPY, 2-8
FORMAT, 2-9
Free Space, 7-4

G

global parameters, 2-2

H

Hashed Tablespace, 1-1
high allocated RBA, 2-7, 2-13
high used RBA, 2-7, 2-13
High-Allocated RBA, 3-3
High-Used RBA, 3-3
The document is an index page containing a list of topics and their page numbers. The entries are organized in a balanced manner, with topics grouped under alphabetically ordered headers. Here is a structured representation of the content:

### I
- IDCAMS DEFINE, 3-18
- image copy, 4-2
- index, 4-2
- INDEX, 2-12
- Index Reorg Advantages, 7-2
- INDEXES, 2-12
- INDEXSPACE, 2-12
- input, 2-1
- Internal IDS Report, 5-1

### K
- KEEPDB2, 3-3

### L
- LOB, 7-8
- LOBMaster, 2-6, 3-8, 7-8
- LOBRECLAIMPERCENT, 2-7, 3-9
- LODPARMS, 7-5
- LPART, 2-5, 2-12

### M
- Mass Delete, 1-1, 1-3
- MASSDELETE, 2-5
- MAXDISORG, 2-6
- MVS JCL SPACE, 3-3
- MXIG, 3-3

### N
- NOCLIX, 2-6
- NOPAD, 2-10

### O
- Object Space Report, 5-2
- online schema change, 7-7

### P
- PAD, 2-10
- Parallel Processing Control, 3-10
- PART, 2-5, 2-6, 2-12
- Partition By Growth (PBG), 2-6, 7-2
- partitions, 1-4, 2-5, 2-12, 2-13
- Pending DDL, 7-7
- percentage of free space, 3-15
- PQTY, 3-3
- PREFORMAT, 2-7, 2-13
- PRIQTY, 9-19
- PSRBD, 7-6

### R
- RBALRSN_CONVERSION, 2-7
- RBDP, 7-6
- Reasons to Not Reorg, 7-2
- Reasons to Reorg, 7-1
- Rebuild Pending, 7-5
- REBUILDICT, 2-8
- RECOVERYDDN, 2-8
- REORG Completion, 6-1
- REORG INDEX, 2-12
- Reorg Pending, 1-3, 7-5
- REORG TABLESPACE, 2-3
- REORP, 7-5, 7-8
- REPORT, 2-10
- REPORTONLY, 2-8, 2-13
- RLSE, 2-9
- Row Counts, 1-4
- ROWFORMAT, 2-6
- RRF, 2-6
- RRGPARMS, 2-1
- RRGPARMX, 2-1
- Runstats, 1-4
- RUNSTATS, 2-2
- RW, 1-2, 1-3

### S
- SAP, 3-7
- sequential datasets, 1-1, 1-3
- SHRLEVEL_CHANGE, 2-8
- SMS, 3-18
- specific parts, 4-1
- Statuses, 1-2
- Summary, 1-1
- Summary Reports, 5-1
- Syntax, 2-1
- Syntax Diagram, 2-3
- SYSADM, 1-1, 1-3
- SYSCOPY, 1-4
- SYSIN, 2-1
- SYSOUT, 2-2
- SYSPUNCH, 2-11
- SYSUTILX, 7-7

### T
- temporal tables, 1-4, 2-10

### U
- UNLOAD, 2-9
- UNLOAD format, 2-8
- USED, 3-3
- Utility ID, 7-3
- UTLPARMS, 2-1
## Index

<table>
<thead>
<tr>
<th>V</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>volsers, 3-18</td>
<td>XRRGRWRO, 6-1</td>
</tr>
<tr>
<td>volume serials, 3-18</td>
<td>XRRGTERM, 6-1</td>
</tr>
<tr>
<td>Volume Serials, 3-18</td>
<td>XSUT0000, 6-2</td>
</tr>
<tr>
<td>VSAM Catalog, 3-3</td>
<td>XSUTALLO, 3-18</td>
</tr>
<tr>
<td>W</td>
<td>XSUTGLOB, 6-2</td>
</tr>
<tr>
<td>WHERE, 2-10</td>
<td>XSUTSYIN, 6-2</td>
</tr>
<tr>
<td>WHERE clause, 1-4</td>
<td>XSUTTERM, 6-2</td>
</tr>
<tr>
<td>WHERE clause., 2-10</td>
<td>XSUTXERR, 6-2</td>
</tr>
<tr>
<td>wildcards, 2-11</td>
<td>XSUTXTND, 3-18</td>
</tr>
</tbody>
</table>