ALTER and CHANGE MANAGER for DB2
Reference Manual

Supporting

Version 12.1.00 of BMC Object Administration for DB2
Version 12.1.00 of BMC Administrative Assistant for DB2
Version 12.1.00 of BMC Database Administration for DB2
Version 12.1.00 of ALTER for DB2
Version 12.1.00 of CHANGE MANAGER for DB2
Version 12.1.00 of BMC Next Generation Technology Database Administration for DB2

December 2016
Contacting BMC Software

Several methods are available for contacting BMC Software.

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

United States and Canada

**Address**

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

**Telephone**

1 713 918 8800

or

1 800 841 2031

**Fax**

1 713 918 8000

Outside United States and Canada

**Telephone**

+01 713 918 8800

**Fax**

+01 713 918 8000

© Copyright 1998 - 2016 BMC Software, Inc.

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

Explorer is a trademark or registered trademark of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

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

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

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

**Restricted rights legend**

U.S. Government Restricted Rights to Computer Software. UNPUBLISHED—RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES. Use, duplication, or disclosure of any data and computer software by the U.S. Government is subject to restrictions, as applicable, set forth in FAR Section 52.227-14, DFARS 252.227-7013, DFARS 252.227-7014, DFARS 252.227-7015, and DFARS 252.227-7025, as amended from time to time. Contractor/Manufacturer is BMC SOFTWARE INC, 2103 CITYWEST BLVD, HOUSTON TX 77042-2827, USA. Any contract notices should be sent to this address.
Customer support

Support website
You can obtain technical support from BMC 24 hours a day, 7 days a week at [http://www.bmc.com/support](http://www.bmc.com/support). From this website, you can:

- Read overviews about support services and programs that BMC offers
- Find the most current information about BMC products
- Search a database for problems similar to yours and possible solutions
- Order or download product documentation
- Download products and maintenance
- Report a problem or ask a question
- Subscribe to receive proactive e-mail alerts
- Find worldwide BMC support center locations and contact information, including e-mail addresses, fax numbers, and telephone numbers

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

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

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

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

- Send an e-mail message to customer_support@bmc.com. (In the Subject line, enter **SupID:**yourSupportContractID, such as SupID:12345.)

- In the United States and Canada, call 1 800 537 1813. Outside the United States and Canada, contact your local support center for assistance.

# Contents

**About this book** ......................................................... 11
Related publications .......................................................... 11
Conventions ........................................................................ 12
Syntax statements .............................................................. 12
Syntax diagrams .................................................................. 13
Summary of changes .......................................................... 14

**Chapter 1** Object types, commands, and action codes 19
DB2 object types .............................................................. 19
Command line commands .................................................. 26
Action codes ...................................................................... 32
Selection List action codes ................................................. 44
Change rules used with object attributes ............................. 45

**Chapter 2** Keywords and variables 53
Keywords for input streams ............................................... 53
  Overview of the keyword tables ....................................... 54
  AEXIN keywords ...................................................... 55
  AEXPIN keywords .................................................... 62
  AJXIN keywords ...................................................... 64
  ALUIN keywords ..................................................... 71
Keywords that override installation options ....................... 139
Symbolic variables for BMC Administrative products .......... 145

**Chapter 3** ALTER and CHANGE MANAGER installation options 159
Installation options .......................................................... 159

**Chapter 4** JCL Generation product options 181
Example of product options ................................................. 181
Descriptions of product option keywords ............................. 189

**Chapter 5** Worklist commands 259
File format ......................................................................... 259
  Multi-line commands .................................................. 260
  Comment lines .......................................................... 261
ALTER and CHANGE MANAGER worklist commands .......... 261
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMS</td>
<td>IDCAMS execution</td>
<td>261</td>
</tr>
<tr>
<td>AUTH</td>
<td>Authorization switching</td>
<td>262</td>
</tr>
<tr>
<td>BASE</td>
<td>Establish baseline</td>
<td>263</td>
</tr>
<tr>
<td>BASF</td>
<td>Mark baseline recoverable</td>
<td>264</td>
</tr>
<tr>
<td>BEGG</td>
<td>Begin global commands</td>
<td>265</td>
</tr>
<tr>
<td>BEGP</td>
<td>Begin parallel processing</td>
<td>265</td>
</tr>
<tr>
<td>BEGU</td>
<td>Begin unit of work</td>
<td>267</td>
</tr>
<tr>
<td>BIND</td>
<td>Bind</td>
<td>267</td>
</tr>
<tr>
<td>BMCB</td>
<td>BMC BASIC UNLOAD utility</td>
<td>268</td>
</tr>
<tr>
<td>BMCC</td>
<td>NGT Copy utility</td>
<td>269</td>
</tr>
<tr>
<td>BMCD</td>
<td>BMC UNLOAD PLUS utility</td>
<td>273</td>
</tr>
<tr>
<td>BMCK</td>
<td>BMC CHECK PLUS utility</td>
<td>280</td>
</tr>
<tr>
<td>BMCL</td>
<td>BMC LOADPLUS utility</td>
<td>280</td>
</tr>
<tr>
<td>BMCR</td>
<td>BMC REORG PLUS utility</td>
<td>293</td>
</tr>
<tr>
<td>BMCS</td>
<td>BMC BMSTATS utility</td>
<td>297</td>
</tr>
<tr>
<td>BMCV</td>
<td>NGT Recover utility</td>
<td>298</td>
</tr>
<tr>
<td>CHEK</td>
<td>IBM CHECK DATA or CHECK LOB utility</td>
<td>305</td>
</tr>
<tr>
<td>CMD</td>
<td>DB2 command</td>
<td>307</td>
</tr>
<tr>
<td>COPY</td>
<td>IBM COPY utility</td>
<td>307</td>
</tr>
<tr>
<td>ENDG</td>
<td>End global commands</td>
<td>309</td>
</tr>
<tr>
<td>ENDP</td>
<td>End parallel processing</td>
<td>310</td>
</tr>
<tr>
<td>ENDU</td>
<td>End unit of work</td>
<td>310</td>
</tr>
<tr>
<td>ERR</td>
<td>Error</td>
<td>310</td>
</tr>
<tr>
<td>GLID</td>
<td>Global authorization ID</td>
<td>311</td>
</tr>
<tr>
<td>GOTO</td>
<td>Bypass command</td>
<td>311</td>
</tr>
<tr>
<td>ISMT</td>
<td>Is table empty</td>
<td>312</td>
</tr>
<tr>
<td>ISMX</td>
<td>Is LOB column empty</td>
<td>313</td>
</tr>
<tr>
<td>JCLP</td>
<td>JCL Generation parameter</td>
<td>314</td>
</tr>
<tr>
<td>LCMD</td>
<td>List command</td>
<td>318</td>
</tr>
<tr>
<td>LDXT</td>
<td>IBM LOAD Extended Text utility</td>
<td>319</td>
</tr>
<tr>
<td>LOAD</td>
<td>IBM LOAD utility</td>
<td>321</td>
</tr>
<tr>
<td>MIGR</td>
<td>Migrate phase</td>
<td>327</td>
</tr>
<tr>
<td>NGTU</td>
<td>NGT utilities</td>
<td>327</td>
</tr>
<tr>
<td>RBLD</td>
<td>IBM REBUILD INDEX utility</td>
<td>329</td>
</tr>
<tr>
<td>REBD</td>
<td>DB2 REBIND</td>
<td>329</td>
</tr>
<tr>
<td>REOR</td>
<td>IBM REORG utility</td>
<td>330</td>
</tr>
<tr>
<td>REXC</td>
<td>REXX and CLIST macro execution</td>
<td>332</td>
</tr>
<tr>
<td>RNAM</td>
<td>Rename object</td>
<td>333</td>
</tr>
<tr>
<td>RNST</td>
<td>IBM RUNSTATS utility</td>
<td>333</td>
</tr>
<tr>
<td>SETA</td>
<td>SET SCHEMA statement</td>
<td>334</td>
</tr>
<tr>
<td>SETP</td>
<td>SET CURRENT PATH statement</td>
<td>335</td>
</tr>
</tbody>
</table>
-SETS (Set current SQL ID) ................................................................. 336
-SPAC (Space estimation source) .......................................................... 337
-SPBX (External SQL stored procedures) ............................................... 338
-SQL (SQL statement) ................................................................. 338
-SQLP (Native SQL stored procedures) ................................................. 341
-SSID (Subsystem ID) ........................................................................ 342
-STOP (Stop) .................................................................................. 343
-SYNC (Sync point) .......................................................................... 343
-TIME (File creation time) .................................................................. 345
-UNLI (IBM UNLOAD utility) ............................................................. 345
-UNRC (Unload record) ...................................................................... 348
-WKID (Work ID) ................................................................................ 350
-ZPRM (ZPARM value) ........................................................................ 351
CM/PILOT worklist commands .............................................................. 351
-ANLY (Analysis) ............................................................................. 352
-BJCL (Batch Execution JCL Generation) .............................................. 353
-COMP (Compare) ........................................................................... 353
-DML (DML section) .......................................................................... 354
-IMP (Import) ................................................................................... 354
-OPTS (Options) ................................................................................ 355
-REPL (REPL section) ........................................................................ 356
-TASK (Task ID) ................................................................................ 357
Other worklist commands ....................................................................... 357
-BMCU (Execute a BMC utility) .......................................................... 357
-DBUG (Debug) .................................................................................. 357
-DSN1 (IBM DSN1COPY utility) ............................................................ 358
-MERG (IBM MERGECOPY utility) ...................................................... 359
-MODI (IBM MODIFY utility) .............................................................. 359
-NOOP (No operation) ........................................................................ 360
-QUI (IBM QUIESCE utility) ............................................................... 360
-REPO (IBM REPORT utility) ............................................................... 361
-REPX (IBM REPAIR utility) ............................................................... 361
-SQLM (Group multiple ALTER statements) ....................................... 361
-STOS (IBM STOSPACE utility) ............................................................ 362
Authorizations used for authorization switching ..................................... 362
Authorization switching using non-GLID processing ............................... 363
Authorization switching using GLID processing ...................................... 367

Chapter 6 Change Definition Language .................................................. 373
File format ........................................................................................ 373
CDL commands .................................................................................. 374
Chapter 7  CM/PILOT Data Manipulation Language (DML)

Actions that you can perform by using DML ............................................................... 557
DML statements .............................................................................................................. 558
Keywords ....................................................................................................................... 559
SET clause .................................................................................................................... 560
WHERE clause ............................................................................................................. 561
DELETE statement ....................................................................................................... 561
DELETE SYNCTABLE statement .................................................................................. 563
DELETE WORKID statement ...................................................................................... 565
LIKE statement ............................................................................................................ 566
MIGRATE statement ................................................................................................... 568
REPLICATE WORKID statement ................................................................................ 573
SET OMITUNICODE statement ................................................................................... 592
SET PRESERVELIMITKEY statement ....................................................................... 593
SET SPACE ESTIMATION statement ......................................................................... 594
UPDATE statement ..................................................................................................... 597
Performance enhancements with the JOIN keyword ................................................. 602
Attributes used in SET and WHERE clauses .............................................................. 604

Contents
About this book

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

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

Related publications

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


- View Quick Course videos (short overviews of selected product concepts, tasks, or features), which are available from the following locations:
  - Documentation Center (primary center and secured center)
  - Support Central (at http://www.bmc.com/support/mainframe-demonstrations)
  - BMC Mainframe YouTube channel (https://www.youtube.com/user/BMCSoftwareMainframe)


Products with online interfaces also offer online Help via the F1 key or, for graphical user interfaces (GUIs), via a Help button.
Tip
If you prefer hardcopy documentation, you can order it from your BMC sales representative or from Support Central. Also, from Support Central you can subscribe to receive proactive e-mail alerts when BMC issues notices.

Conventions

This document uses the following special conventions:

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

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

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

Syntax statements

This topic explains conventions for showing syntax statements.

A sample statement follows:

```
COMMAND KEYWORD1 [KEYWORD2 | KEYWORD3] KEYWORD4={YES | NO} fileName...
```

The following table explains conventions for syntax statements and provides examples:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Example</th>
</tr>
</thead>
</table>
| Items in italic type represent variables that you must replace with a name or value. If a variable is represented by two or more words, initial capitals distinguish the second and subsequent words. | alias
databaseDirectory
serverHostName |
## Convention

Brackets indicate optional items. Do not type the brackets when you enter the option. A comma means that you can choose one or more of the listed options. You must use a comma to separate the options if you choose more than one option.

```
[tableName, columnName, field]
[-full, -incremental, -level]
```

Braces indicate that at least one of the enclosed items is required. Do not type the braces when you enter the item.

```
{DBDName | tableName}
UNLOAD device={disk | tape, fileName | deviceName}
{-a | -c}
```

A vertical bar means that you can choose only one of the listed items. In the example, you would choose either `commit` or `cancel`.

```
{commit | cancel}
```

An ellipsis indicates that you can repeat the previous item or items as many times as necessary.

```
columnName...
```

## Syntax diagrams

The following figure shows the standard format for syntax diagrams:

![Syntax Diagram](image)

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

The following guidelines provide additional information about syntax diagrams:

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

## Summary of changes

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

### Version 12.1.00, December 2016

This release fixes known problems and includes the following enhancements:

- Support for DB2 Version 12.1 features
— Support for APPLCOMPAT V12R1Mnnn in native stored procedures

— Support for LOB compression

DB2 Version 12 lets you compress LOB table spaces where the base table is in a universal table space. CHANGE MANAGER supports this feature, and lets you specify whether a LOB is compressed in the Create Auxiliary Tablespace and Auxiliary Tablespace Detail panels.

— Support for BUSINESS_TIME enhancement

IBM DB2 Version 12 supports the INCLUSIVE and EXCLUSIVE option for the BUSINESS_TIME parameter for ALTER TABLE and CREATE TABLE statements. ALTER and CHANGE MANAGER support this option.

— Support for online index compression

ALTER and CHANGE MANAGER support the changed behavior of DB2 when you alter the compression value of an index on a universal table space. DB2 Version 12 places the index into Advisory REORG pending (AREOR) status.

— PREVENT_NEW_IXCTRL_PART ZPARM

The Analysis component of CHANGE MANAGER now verifies the PREVENT_NEW_IXCTRL_PART ZPARM when you need to LIKE, migrate, or rebuild an index-controlled partitioned table space, or when you need to include the table space in a full-recovery baseline.

— ALTER and CHANGE MANAGER support the new value of ALWAYS_PENDING for the DDL_MATERIALIZATION ZPARM.

Version 12.1 of ALTER and CHANGE MANAGER tolerate the following features of IBM DB2 Version 12.

— The extension of range-partitioned table spaces with the PAGENUM RELATIVE attribute of the CREATE TABLESPACE and ALTER TABLESPACE statements

— Enhanced triggers

— The TRANSFER OWNERSHIP command

— Foreign keys with the PERIOD BUSINESS_TIME clause

— Inserting partitions into a range-partitioned table space

— INSERT ALGORITHM clause in CREATE TABLESPACE and ALTER TABLESPACE statements

— Unicode columns in an EBCDIC table
Automatic change rules for stored procedures
CHANGE MANAGER can now apply an automatic change rule for the version number of a stored procedure. This enables you to specify stored procedures in a DB2 catalog to DB2 catalog comparison.

Support for spanned records
CHANGE MANAGER can now unload and load existing LOB and XML column data by using spanned records.

Support for NGT Reorg
You can now specify NGT Reorg to reorganize objects on DB2 Version 10 and 11 subsystems.

Support for PLANMGMT
This release adds support for the PLANMGMT rebind option. You can now specify the PLANMGMT value that ALTER and CHANGE MANAGER use during an rebind operation.

Toleration of IDAA tables
The products recognize when a table has been added to IDAA and do not change the definition of the table.

Deprecation of the LOB DATA MOVER program
The LOB DATA MOVER program is no longer required to move LOB data, and has been deprecated. This was a feature of the BMC Database Administration for DB2 solution.

RECOVER PLUS for DB2 and COPY PLUS for DB2 name change
Starting with this release, the name of the RECOVER PLUS for DB2 product has changed to the BMC Next Generation Technology Recover for DB2 for z/OS (or NGT Recover) product. The name of the COPY PLUS for DB2 product has changed to the BMC Next Generation Technology Copy for DB2 for z/OS (or NGT Copy) product.

Support for IBM DB2
ALTER and CHANGE MANAGER now support the following versions and modes of DB2:

- DB2 V12 Function level 100
- DB2 V12 Function level 500
- DB2 V11 Conversion mode (CM)
- DB2 V11 Enabling-new-function mode (ENFM)
— DB2 V11 New-function mode (NFM)
— DB2 V10 NFM
Object types, commands, and action codes

This section lists the object types, commands, action codes, and change rules that you can use in ALTER and CHANGE MANAGER.

DB2 object types

The Specification component of ALTER and CHANGE MANAGER uses a mixed list to display DB2 object types and object names.

Table 1 on page 19 shows the abbreviations that Specification uses, and the data types for each part of the object names (in bytes).

Table 1: DB2 object types

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
<th>Data type of object name (in bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Data type (part 1)</td>
</tr>
<tr>
<td>AL</td>
<td>Alias</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>CK</td>
<td>Check constraint (check)</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>DB</td>
<td>Database</td>
<td>VARCHAR(24) a</td>
</tr>
<tr>
<td>FK</td>
<td>Foreign key</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IX</td>
<td>Index</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td></td>
<td>Auxiliary index</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td></td>
<td>XML index</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXC</td>
<td>Clustering index</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXA</td>
<td>Orphaned auxiliary index</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
<td>Data type of object name (in bytes)</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
<td>------------------------------------</td>
</tr>
</tbody>
</table>
| IXCT         | Clustering index  
This partitioned index is created for a table that uses table-controlled partitioning. | VARCHAR(128) | VARCHAR(128) | NA |
| IXCT2        | Clustering index using the PAGENUM parameter (PBR2 Index)  
This partitioned index is created for a range-partitioned (PBR) table space. | VARCHAR(128) | VARCHAR(128) | NA |
| IXG          | Unique index that is defined as ROWID GENERATED BY DEFAULT | VARCHAR(128) | VARCHAR(128) | NA |
| IXGC         | Unique and clustering index that is defined as ROWID GENERATED BY DEFAULT | VARCHAR(128) | VARCHAR(128) | NA |
| IXGCT        | Unique and clustering index that is defined as ROWID GENERATED BY DEFAULT  
This partitioned index is created for a table that uses table-controlled partitioning. | VARCHAR(128) | VARCHAR(128) | NA |
| IXGT         | Unique index that is defined as ROWID GENERATED BY DEFAULT  
This index is created for a table that uses table-controlled partitioning. | VARCHAR(128) | VARCHAR(128) | NA |
<p>| IXN          | Node ID index for an XML base table | VARCHAR(128) | VARCHAR(128) | NA |
| IXP          | Primary index | VARCHAR(128) | VARCHAR(128) | NA |
| IXPC         | Primary and clustering index | VARCHAR(128) | VARCHAR(128) | NA |</p>
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
<th>Data type of object name (in bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Data type (part 1)</td>
</tr>
</tbody>
</table>
| IXPCT        | Primary and clustering index  
This partitioned index is created for a table that uses table-controlled partitioning. | VARCHAR(128) | VARCHAR(128) | NA |
| IXPT         | Primary index  
This partitioned index is created for a table that uses table-controlled partitioning. | VARCHAR(128) | VARCHAR(128) | NA |
| IXR          | Unique index that enforces the uniqueness of a nonprimary parent key; referenced by a foreign key | VARCHAR(128) | VARCHAR(128) | NA |
| IXRC         | Unique and clustering index that enforces the uniqueness of a nonprimary parent key | VARCHAR(128) | VARCHAR(128) | NA |
| IXRCT        | Unique and clustering index that enforces the uniqueness of a nonprimary parent key  
This partitioned index is created for a table that uses table-controlled partitioning. | VARCHAR(128) | VARCHAR(128) | NA |
| IXRT         | Unique index that enforces the uniqueness of a nonprimary parent key  
This partitioned index is created for a table that uses table-controlled partitioning. | VARCHAR(128) | VARCHAR(128) | NA |
<p>| IXS          | Index on an expression | VARCHAR(128) | VARCHAR(128) | NA |
| IXST         | Partitioned index on an expression that is created for a table that uses table-controlled partitioning | VARCHAR(128) | VARCHAR(128) | NA |
| IXSU         | Unique index on an expression | VARCHAR(128) | VARCHAR(128) | NA |</p>
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
<th>Data type of object name (in bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Data type (part 1)</td>
</tr>
<tr>
<td>IXT</td>
<td>Partitioned index that is created for a table that uses table-controlled partitioning</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXU</td>
<td>Unique index</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXUC</td>
<td>Unique and clustering index</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXUCT</td>
<td>Unique and clustering index This partitioned index is created for a table that uses table-controlled partitioning.</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXUT</td>
<td>Unique index This partitioned index is created for a table that uses table-controlled partitioning.</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXV</td>
<td>Index that is defined on an XML column in a base table (TBB or TBBT) This object code replaces the IXX object code.</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXVD</td>
<td>DocID index that is defined on an XML column in a base table (TBB or TBBT) DB2 implicitly creates this index. This object code replaces the IXXD object code.</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXVN</td>
<td>NodeID index that is defined on an XML column in a base table (TBB or TBBT)</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXVU</td>
<td>Unique XML index</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXW</td>
<td>Unique index that is defined as WHERE NOT NULL</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>IXWC</td>
<td>Unique and clustering index that is defined as WHERE NOT NULL</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
<td>Data type of object name (in bytes)</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>SG</td>
<td>Storage group (Stogroup)</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>SP</td>
<td>Stored procedure</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>SPE</td>
<td>External stored procedure</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>SPN</td>
<td>Native SQL stored procedure</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>SPNA</td>
<td>Active version of native SQL stored procedure</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>SPS</td>
<td>External SQL stored procedure</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>SQ</td>
<td>Sequence</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>SY</td>
<td>Synonym</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>TB</td>
<td>Table</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>TBB</td>
<td>Table containing a large object (LOB) or an XML column</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>TBH</td>
<td>Table that DB2 uses to store historical versions of rows from the</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>TBBH</td>
<td>associated system-period temporal table</td>
<td></td>
</tr>
<tr>
<td>TBBT</td>
<td>Base table that uses table-controlled partitioning</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>TBC</td>
<td>Clone table</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>TBGT</td>
<td>Global temporary table</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>TBQ</td>
<td>Materialized query table (MQT)</td>
<td>VARCHAR(128)</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
<td>Data type of object name (in bytes)</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td><strong>TBQT</strong></td>
<td>Materialized query table (MQT) that uses table-controlled partitioning</td>
<td>VARCHAR(128) VARCHAR(128) NA</td>
</tr>
<tr>
<td><strong>TBR</strong></td>
<td>Archive table Archive table for table containing a large object (LOB) or an XML column (base table)</td>
<td>VARCHAR(128) VARCHAR(128) NA</td>
</tr>
<tr>
<td><strong>TBT</strong></td>
<td>Table that uses table-controlled partitioning</td>
<td>VARCHAR(128) VARCHAR(128) NA</td>
</tr>
<tr>
<td><strong>TR</strong></td>
<td>Trigger in which the trigger text references the table</td>
<td>VARCHAR(128) VARCHAR(128) NA</td>
</tr>
<tr>
<td><strong>TRA</strong></td>
<td>Active advanced trigger</td>
<td>VARCHAR(128) VARCHAR(128) VARCHAR(64)</td>
</tr>
<tr>
<td><strong>TRAB</strong></td>
<td>Active advanced trigger that references the table</td>
<td>VARCHAR(128) VARCHAR(128) VARCHAR(64)</td>
</tr>
<tr>
<td><strong>TRB</strong></td>
<td>Trigger that references the table</td>
<td>VARCHAR(128) VARCHAR(128) NA</td>
</tr>
<tr>
<td><strong>TS</strong></td>
<td>Table space Auxiliary table space XML table space The TS might have any of the following characteristics:</td>
<td>VARCHAR(24) VARCHAR(24) NA</td>
</tr>
<tr>
<td></td>
<td>■ Nonpartitioned</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Created with the CTB command-line command or action code</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Imported without a dependent table</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Partitioned and incomplete</td>
<td></td>
</tr>
<tr>
<td><strong>TSA</strong></td>
<td>Orphaned auxiliary table space</td>
<td>VARCHAR(24) VARCHAR(24) NA</td>
</tr>
<tr>
<td><strong>TSG</strong></td>
<td>Partition-by-growth table space</td>
<td>VARCHAR(24) VARCHAR(24) NA</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
<td>Data type of object name (in bytes)</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data type (part 1) Data type (part 2) Data type (part 3)</td>
</tr>
</tbody>
</table>
| TSI          | Partitioned table space that contains a table that uses index-controlled partitioning  
A TSI can be any TS that is LIKEd from a TSI or any TS that is converted to index-controlled partitioning. | VARCHAR(24) a VARCHAR(24) c NA |
| TST          | Partitioned table space that contains a table that uses table-controlled partitioning  
A TST can be any TS that is LIKEd from a TST or any TS that is converted to table-controlled partitioning. | VARCHAR(24) a VARCHAR(24) c NA |
| TSR          | A range-partitioned table space | VARCHAR(24) a VARCHAR(24) c NA |
| TSR2         | Partitioned table space that contains a table that uses range partitioning with the PAGENUM parameter | VARCHAR((24) a VARCHAR(24) a NA |
| UC           | Unique constraint | VARCHAR(128) VARCHAR(128) VARCHAR(128) |
| UCP          | Primary key constraint | VARCHAR(128) VARCHAR(128) VARCHAR(128) |
| UCU          | Unique (nonprimary) key constraint | VARCHAR(128) VARCHAR(128) VARCHAR(128) |
| VW           | View | VARCHAR(128) VARCHAR(128) NA |

a In a Unicode subsystem, the database name can be defined as VARCHAR(24). However, in a non-Unicode subsystem, the name can be defined only as VARCHAR(8).

b The synonym owner is limited to a maximum of 8 bytes.

c In a Unicode subsystem, the table space name can be defined as VARCHAR(24). However, in a non-Unicode subsystem, the name can be defined only as VARCHAR(8).
Command line commands

Every panel has a Command line. The Command line usually appears directly beneath the title bar, although its position depends on your ISPF profile.

You can issue Command line commands to:

- Navigate among panels without using the menus
- Display BMC environment information
- Modify the appearance of panels
- Modify the function of the components of ALTER and CHANGE MANAGER
- Display any BMC Administrative or Utility product message

You can enter the commands listed in Table 2 on page 26 in uppercase or lowercase letters. Use the HELP command (F1) from any panel to see what commands are available from that panel. Some panels list frequently used commands at the bottom of the panel.

Table 2: Command line commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>Displays the Archive Table Information panel</td>
</tr>
<tr>
<td>ALTER</td>
<td>Displays the Create, Alter, or Drop DB2 Objects panel You can issue this command only from the Main Menu.</td>
</tr>
<tr>
<td>ATTR</td>
<td>Displays the panel attributes panel, from which you can change screen attributes and attributes for viewing objects with names that are too long to be displayed</td>
</tr>
<tr>
<td>ATTR RESET</td>
<td>Resets the screen colors to the default colors</td>
</tr>
<tr>
<td>AX</td>
<td>Displays a list of the auxiliary objects that are associated with the large object (LOB) columns in a base table (TBB) You can issue this command only from the Table Detail Panel (for a base table) or the Table Column Detail panel (for a LOB column).</td>
</tr>
<tr>
<td>BROWSE</td>
<td>Invokes the ISPF BROWSE function on a named diagnostic data set You cannot use the BROWSE command if you specify TERM or SYSOUT for diagnostics. You can issue this command only from a panel in which a data set is named.</td>
</tr>
<tr>
<td>CAL CR AL</td>
<td>Creates an alias</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>CANcel</td>
<td>Exits the current panel without saving changes</td>
</tr>
<tr>
<td>CAX</td>
<td>Creates an auxiliary table space, table, and index for an existing base table from the Table Column Detail panel</td>
</tr>
<tr>
<td>CCK CR CK</td>
<td>Creates a check constraint</td>
</tr>
<tr>
<td>CDB CR DB</td>
<td>Creates a database</td>
</tr>
<tr>
<td>CFK CR FK</td>
<td>Creates a foreign key</td>
</tr>
</tbody>
</table>
| CGT CR GT | Creates a global temporary table  
In the Mixed List panel, this command is available for an existing table or base table or from the Command line. |
| CH | Displays a changed object list from an object Specification panel or Mixed List panel  
Orphaned auxiliary indexes are displayed in the Unattached Objects Section of the list. |
<p>| CIX CR IX | Creates an index or an auxiliary index for a large object (LOB) column in an existing base table |
| CM | Displays a comment panel for a table, base table, auxiliary table, table column, view, view column, alias, index, or auxiliary index from the appropriate detail or auxiliary template panel |
| CO | Displays a column list for the object from an object detail panel |
| CReate | Creates DB2 objects from the Mixed List panel, Changed Objects List panel, detail panel, or create panel |
| CSG CR SG | Creates a storage group |
| CSP CR SP | Creates an external stored procedure or a native SQL stored procedure |
| CSQ CR SQ | Creates a sequence |
| CSY CR SY | Creates a synonym |
| CTB CR TB | Creates a table or an auxiliary table for a large object (LOB) column in an existing auxiliary table space |</p>
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>
| CTC     | Creates a clone table of an existing table  
This command is not valid on materialized query tables (MQTs),  
global temporary tables, other clone tables, or system-period  
temporal tables. |
| CTR     | Creates a trigger  
CR TR    |
| CTS     | Creates a table space or an auxiliary table space for a large object  
(LOB) column in an existing base table  
CR TS    |
| CUC     | Creates a unique or primary constraint  
CR UC    |
| CVW     | Creates a view  
CR VW    |
| DOPTS   | Displays a list of the keywords and values in the installation  
options module in an ISPF edit session  
This list does not include the installation option keywords that  
have been replaced by keywords in the product options file  
(POF).  
You can assemble the installation options in the session to create  
an additional installation options module. |
| DROPALL | Drops all objects in the list  
Objects are selected to be dropped if the Act column is blank or  
marked ‘$’ for space estimation only. In the Mixed List panel,  
this command also applies to orphaned auxiliary objects. If an  
auxiliary object has been LIKEd, this command removes the  
requested changes for the object. That is, the command performs  
the actions of an UNDOALL command. |
| DV      | Displays the Table Column Default Value panel from the Table  
Column Detail panel |
| END     | Exits the current panel, saves any changes, and displays the next  
panel in sequence |
| ENVI    | Displays the BMC Environment panel that contains information  
such as product name, version, installation options module  
name, ship date, and plan names  
ENVIRONMENT |
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>
| **FInd** | Locates character strings in a scrollable list  
The syntax of the command is as follows:  
**FInd[ [string,'string', or "string"]][ NEXT, FIRST, LAST, or PREV]**  
In the preceding syntax line:  
- Keywords are shown in uppercase characters.  
- All parameters are optional.  
- NEXT is the default parameter.  
The parameters are described as follows:  
- **string**  
The string of characters to find. If the string contains spaces, it must be enclosed with single quotation marks. If the string contains single quotation marks, it must be enclosed with double quotation marks. If the *string* parameter is specified, it must be the first parameter.  
- **NEXT**—finds the next occurrence of the string  
- **FIRST**—finds the first occurrence of the string  
- **LAST**—finds the last occurrence of the string  
- **PREV**—finds the previous occurrence of the string  
If you issue the FIND command with no parameters, the previously found string is located if it exists. If no previously found string exists, the cursor is located at the first item in the list, as displayed on the panel. |
<p>| <strong>FP</strong> | Displays the Table Column Fieldproc Parms panel from a Table Column Detail panel |
| <strong>HELP</strong> | Displays help information |
| <strong>HI</strong> | Displays the History Table Information panel |
| <strong>ICP</strong> | Converts a table to index-controlled partitioning from the Table Detail panel |
| <strong>ID</strong> | Displays the Identity Column Detail panel from the Table Column Detail panel |
| <strong>JCLDEBUG OFF/ON</strong> | Turns foreground JCL Generation debugging off or on |</p>
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KY</td>
<td>Displays the Index Keys List panel from an Index Detail panel. For a table that uses table-controlled partitioning, you can type KY to display the Partitioning Key Mixed List panel from the Table Detail panel. This command is not valid for an index that is defined on an XML column.</td>
</tr>
<tr>
<td>LIX</td>
<td>Displays a list of the auxiliary indexes that are associated with a large object (LOB) column from a column detail panel.</td>
</tr>
<tr>
<td>LK</td>
<td>Displays the Limit Key Detail panel from an Index Partition Detail panel (for index-controlled partitioning) or the Tablespace Partition Detail panel. For a table that uses table-controlled partitioning, you can type LK on the Table Detail panel or the Tablespace Detail panel to display the Tablespace Parts List panel.</td>
</tr>
<tr>
<td>LTB</td>
<td>Displays a list of the auxiliary tables that are associated with a large object (LOB) column from a column detail panel.</td>
</tr>
<tr>
<td>LTS</td>
<td>Displays a list of the auxiliary table spaces that are associated with a large object (LOB) column from a column detail panel.</td>
</tr>
<tr>
<td>MAINT</td>
<td>Displays the Product Maintenance List panel that contains information about fixes (PTFs) that you have applied to the product and the product components. You can issue this command only from the BMC Environment panel.</td>
</tr>
<tr>
<td>MAX</td>
<td>Displays the maximum assigned value for an identity column in the Identity Column Detail panel, or for a sequence in the Create Sequence panel or Sequence Detail panel.</td>
</tr>
<tr>
<td>MIGALL</td>
<td>Selects all objects, except orphaned auxiliary objects, in the mixed list for migration. Objects are selected for migration if the Act column is blank or marked ‘S’ for space estimation only.</td>
</tr>
<tr>
<td>MIGRATE</td>
<td>Displays the Migrate DB2 Objects Using Specification panel. You can issue this command only from the Main Menu.</td>
</tr>
<tr>
<td>MP</td>
<td>Displays the parameters for a materialized query table (MQT) from the MQT Detail or Create MQT panel.</td>
</tr>
<tr>
<td>MQT</td>
<td>Converts a table to a materialized query table (MQT) from the Table Detail panel, or creates an MQT from the Create Table panel. On the View Detail panel, you can issue this command to create an MQT that is based on the definition of the view.</td>
</tr>
<tr>
<td>MT</td>
<td>Displays text for a materialized query table (MQT) from the Create MQT, MQT Detail, Create MQT Parameters, or MQT Parameters panel.</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>PA</td>
<td>Displays the path for a trigger, view, or check constraint</td>
</tr>
<tr>
<td>PEALL</td>
<td>Propagates the estimated space to the primary quantity (PRIQTY) for all of the partitions in an index in the Index Space Estimation panel or in a table space in the Tablespace Estimation (Partitioned) panel</td>
</tr>
<tr>
<td>PCO</td>
<td>Displays the Parent Key Columns panel from the Create Foreign Key panel. This command is valid only when the parent table’s primary key is not used.</td>
</tr>
<tr>
<td>PK</td>
<td>Displays the Constraint Key Mixed List panel from the Table Detail panel or Table Columns List panel</td>
</tr>
<tr>
<td>POP</td>
<td>Toggles pop-up message boxes on or off</td>
</tr>
<tr>
<td>PREVIOUS</td>
<td>Exits the current panel and displays the previous panel</td>
</tr>
<tr>
<td>PT</td>
<td>Displays an object partition or attribute list panel from a detail or template panel. For a table that uses table-controlled partitioning, you can type PT on the Table Detail panel or the Tablespace Detail panel to display the Tablespace Parts List panel.</td>
</tr>
<tr>
<td>SE</td>
<td>Displays space estimation information from a Tablespace Detail or Index Detail panel</td>
</tr>
<tr>
<td>SORT</td>
<td>Displays the Sort Specifications panel, from which you can specify the columns on which to sort a list of work IDs. You can specify the sequence of the columns and the order of the data in the columns. You can also sort on a single column by placing the cursor within the heading for that column and pressing Enter. To return to the original sort order, issue the SORT command. If you specify a sequence for the Stat Date column, you cannot specify a sequence for the Stat YYYY, Stat MM, or Stat DD column.</td>
</tr>
<tr>
<td>TCP</td>
<td>Converts a table to table-controlled partitioning from the Table Detail panel</td>
</tr>
<tr>
<td>TSO POFRESET</td>
<td>Resets all of the ISPF variables in the ISPF profile with the variables in the initial POF. TSO POFRESET POF(dataSetName(POFMember)) resets all of the ISPF variables in the ISPF profile with the variables in the specified initial or user POF name. The POFRESET CLIST enables these commands. The CLIST is located in the HLQ.DBCLLIB library.</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>TX</td>
<td>Displays the view text, trigger text, check constraint text, materialized query table (MQT) text, index on an expression, or index defined on an XML column from the Mixed List panel, a Detail panel, or a Create panel. The product formats some of the text for all of the objects, except for indexes on expressions and indexes defined on XML columns.</td>
</tr>
<tr>
<td>UNDOALL</td>
<td>Removes the requested changes for all of the objects on the list. In the Mixed List panel, this command also applies to orphaned auxiliary objects.</td>
</tr>
<tr>
<td>VIEWX</td>
<td>Displays the execution status of a work ID from the Execution JCL Processing Interface panel.</td>
</tr>
<tr>
<td>VL</td>
<td>Displays the Volume List and SMS Classes panel for VCAT-defined table spaces from the Auxiliary Tablespace Attribute Detail panel or the Auxiliary Index Attribute Detail panel.</td>
</tr>
<tr>
<td>W</td>
<td>Enables you to edit the worklist for a work ID from the WORKID Selection List panel. The work ID must have a status of Analyzed, Exec Strt, or Exec Comp. In addition, the work ID must not be used with an outbound migrate profile that specifies multiple locations, because the profile can be used to generate multiple worklists.</td>
</tr>
<tr>
<td>XP</td>
<td>Displays the parameters for an index that is defined on an XML column from the Index Detail or Create Index panel.</td>
</tr>
</tbody>
</table>

### Action codes

Many of the panels in the products contain a scrollable list of items. When the panel contains an **Act** column in the list, you can enter an action code (or line command) in the field adjacent to the object on which you want to take an action.

Not all action codes are valid for every object on a panel. The **Use** column in Table 3 on page 33 lists the objects on which an action code can be used.

---

**Note**

If you press the END key while action codes are pending, the codes are executed before the panel terminates.

---

32  *ALTER and CHANGE MANAGER for DB2 Reference Manual*
Table 3: Action codes

<table>
<thead>
<tr>
<th>Action code</th>
<th>Definition</th>
<th>Description</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>After</td>
<td>Copies or moves a key to the column after a particular column in a list of columns</td>
<td>TS, TB, FK, IX, UC, CK, TR, VW</td>
</tr>
<tr>
<td>A</td>
<td>Analyze</td>
<td>Invokes the Analysis Interface panels for a work ID</td>
<td>Work ID</td>
</tr>
<tr>
<td>AL</td>
<td>Aliases</td>
<td>Displays the aliases of a dependent object</td>
<td>SG, DB, TS, TB, GT, VW</td>
</tr>
<tr>
<td>AX</td>
<td>Auxiliary object (LOB)</td>
<td>Displays the auxiliary objects for large object (LOB) columns that are associated with a base table (TBB)</td>
<td>TBB, TBB column</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In the Table Columns List panel, this command can be used if the data type of the column is defined as a large object (LOB) (such as a binary large object, or BLOB; character large object, or CLOB; or double-byte character large object, or DBCLOB) or as a distinct or user-defined type (UDT) that is based on a LOB data type.</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>Before</td>
<td>Copies or moves a key to the column before a particular column in a list of columns</td>
<td>TS, TB, FK, IX, UC, CK, TR, VW</td>
</tr>
<tr>
<td>B</td>
<td>Browse</td>
<td>Browses an object</td>
<td>Work ID, baseline, baseline profile, migrate profile, task ID, application, script</td>
</tr>
<tr>
<td>C</td>
<td>Copy (CHANGE MANAGER object)</td>
<td>Copies a migrate or baseline profile that is displayed in the object Profile Selection List panel</td>
<td>Baseline profile, migrate profile, task ID, script</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You must change the profile name after copying the object.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>For a CM/PILOT task ID or script, this code creates a new task ID, script, or step in a script by using the current object as a template.</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>Copy (DB2 object)</td>
<td>Copies a key or column from a list of columns</td>
<td>TS, TB, FK, IX, UC, CK, TR, VW</td>
</tr>
<tr>
<td>C</td>
<td>Create</td>
<td>Creates an object of the same type</td>
<td>SG, DB, TS, TB, GT, CK, FK, IX, VW, AL, SY, UC, SQ</td>
</tr>
<tr>
<td>Action code</td>
<td>Definition</td>
<td>Description</td>
<td>Use</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>CAL</td>
<td>Create alias</td>
<td>Creates an alias</td>
<td>TB, GT, VW</td>
</tr>
<tr>
<td>CAX</td>
<td>Create auxiliary object</td>
<td>Displays the create auxiliary object template panel, from which you can create auxiliary objects.</td>
<td>TBB column</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In the Table Columns List panel, this command can be used if the data type of the column is defined as a large object (LOB) (such as a binary large object, or BLOB; character large object, or CLOB; or double-byte character large object, or DBCLOB) or as a distinct or user-defined type (UDT) that is based on a LOB data type. This command cannot be used if you are liking or creating a TBB. It can, however, be used when you edit the TBB after it has been stored in the CD tables.</td>
<td></td>
</tr>
<tr>
<td>CC</td>
<td>Block copy</td>
<td>Copies a block of keys or columns from a list of columns</td>
<td>TS, TB, FK, IX, UC, CK, TR, VW</td>
</tr>
<tr>
<td>CCK</td>
<td>Create check constraint</td>
<td>Creates a check constraint</td>
<td>TB</td>
</tr>
<tr>
<td>CDB</td>
<td>Create database</td>
<td>Creates a database</td>
<td>SG</td>
</tr>
<tr>
<td>CFK</td>
<td>Create foreign key</td>
<td>Creates a foreign key</td>
<td>TB</td>
</tr>
<tr>
<td>CGT</td>
<td>Create global temporary table</td>
<td>Creates a global temporary table</td>
<td>TB</td>
</tr>
<tr>
<td>CIX</td>
<td>Create index</td>
<td>Creates an index or an auxiliary index for a large object (LOB) column on an existing base table (TBB)</td>
<td>TB, TBB column, aux TB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>When creating auxiliary indexes, this command can be used in the Table Columns List panel if the data type of the column is defined as a large object (LOB) (such as a binary large object, or BLOB; character large object, or CLOB; or double-byte character large object, or DBCLOB) or as a distinct or user-defined type (UDT) that is based on a LOB data type. This command cannot be used if you are liking or creating a TBB. It can, however, be used when you edit the TBB after it has been stored in the CD tables.</td>
<td></td>
</tr>
<tr>
<td>CK</td>
<td>Check constraint</td>
<td>Displays the check constraint for a table</td>
<td>CK</td>
</tr>
<tr>
<td>Action code</td>
<td>Definition</td>
<td>Description</td>
<td>Use</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>CM</td>
<td>Comment</td>
<td>Displays a comment panel for an object or an auxiliary object</td>
<td>TB, TBB, TBB column, TBQ, GT, IX, VW, TR, AL, VW, SP, SQ</td>
</tr>
<tr>
<td>CO</td>
<td>Column</td>
<td>Displays the columns in a table or a view, the index key columns and table columns in an index, or the constraint key columns and parent table columns in a primary or unique key constraint</td>
<td>TB, TBB, TBQ, GT, UCU, UCP, VW, IX</td>
</tr>
<tr>
<td>CSY</td>
<td>Create synonym</td>
<td>Creates a synonym</td>
<td>TB, GT, VW</td>
</tr>
<tr>
<td>CT</td>
<td>Constraint text</td>
<td>Displays the text associated with a check constraint</td>
<td>CK</td>
</tr>
<tr>
<td>CTB</td>
<td>Create table</td>
<td>Creates a table or an auxiliary table for a large object (LOB) column</td>
<td>TS, GT, TBB column, aux TB</td>
</tr>
<tr>
<td>CTC</td>
<td>Create clone table</td>
<td>Creates a clone table of an existing table</td>
<td>TB, TBB, TBBT, TBT</td>
</tr>
<tr>
<td>CTR</td>
<td>Create trigger</td>
<td>Creates a trigger</td>
<td>TB</td>
</tr>
<tr>
<td>Action code</td>
<td>Definition</td>
<td>Description</td>
<td>Use</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>CTS</td>
<td>Create table space</td>
<td>Creates a table space or an auxiliary table space for a large object (LOB) column. When creating auxiliary table spaces, this command can be used in the Table Columns List panel if the data type of the column is defined as a large object (LOB) (such as a binary large object, or BLOB; character large object, or CLOB; or double-byte character large object, or DBCLOB) or as a distinct or user-defined type (UDT) that is based on a LOB data type. This command cannot be used if you are liking or creating a base table (TBB). It can, however, be used when you edit the TBB after it has been stored in the CD tables.</td>
<td>DB, TBB column</td>
</tr>
<tr>
<td>CUC</td>
<td>Create unique constraint</td>
<td>Creates a unique or primary constraint</td>
<td>TB, IX</td>
</tr>
<tr>
<td>CVW</td>
<td>Create view</td>
<td>Creates a view</td>
<td>TB, GT, VW</td>
</tr>
<tr>
<td>D</td>
<td>Delete</td>
<td>Deletes an object or a step in a script. Use a number in conjunction with this action code to apply the action multiple times.</td>
<td>Work ID, baseline, baseline profile, migrate profile, task ID, application, script</td>
</tr>
<tr>
<td>Action code</td>
<td>Definition</td>
<td>Description</td>
<td>Use</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| D          | Drop                               | On a list panel, marks an object or an auxiliary object for dropping
You cannot drop an altered object. To drop an altered object, undo the alter and then drop.
Altering data structures (by using an alter-type work ID) causes the worklist to drop the object. Migrating data structures (by using a migrate-type work ID) does not drop the object but does exclude it from the migration.
In the Mixed List panel, an orphaned auxiliary index cannot be dropped; instead, the requested changes for the index are removed.
In the Auxiliary Tablespace List panel, an auxiliary table space cannot be dropped if it contains an auxiliary table or if the base table (TBB) is not dropped. An auxiliary table cannot be dropped if it contains data. An auxiliary index cannot be dropped unless it is empty or the TBB is dropped. | SG, DB, TS, TSA, TS partition, TB, TBB, TBB column, TB column, GT, CK, FK, IX, IX partition, IXA, IXUC, IXP, UCP, IXU, IXG, VW, TR, AL, SY, SP, SQ |
| DB         | Database                            | Displays the databases of a dependent object                                                                                                                                                                                                                                         | SG                                                                 |
| DD         | Block delete                        | Deletes a block of keys or columns from a list of columns                                                                                                                                                                                                                          | TS, TB, FK, IX, UC, CK, TR, VW                                      |
| DV         | Default value                       | In the Table Columns List panel or the Table Column Detail panel, displays a default value panel from which you can specify a new default value for a column                                                                                                                                 | TB column                                                          |
| E          | Edit (DB2 object)                   | Displays the detail panel for the selected object or auxiliary object on which you can specify change requests
If you use the CAX command to create an auxiliary object, using the E action code in an auxiliary object list panel displays a create auxiliary object panel.                                                                                                                                 | SG, DB, TS, TS partition, TB, TBB, TBQ, GT, CK, FK, IX, IX partition, VW, TR, AL, SY, SP, SQ |
<p>| E          | Edit (CHANGE MANAGER object)        | Enables you to edit an object or a step in a script                                                                                                                                                                                                                             | Work ID, baseline profile, migrate profile, task ID, application, script |
| E          | Edit comment                        | Enables you to edit the comments for a baseline                                                                                                                                                                                                                                | Baseline                                                          |</p>
<table>
<thead>
<tr>
<th>Action code</th>
<th>Definition</th>
<th>Description</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>FK</td>
<td>Foreign keys</td>
<td>Displays the foreign keys of the specified table</td>
<td>SG, DB, TS, TB</td>
</tr>
<tr>
<td>FP</td>
<td>Fieldproc parameters</td>
<td>In the Table Columns List panel, displays the Table Column FieldprocParms panel, from which you can specify a field procedure exit routine and a parameter list for a column.</td>
<td>TB column</td>
</tr>
<tr>
<td>G</td>
<td>Generate CDL</td>
<td>Displays the Compare CDL Options panel, where CDL can be generated</td>
<td>Work ID</td>
</tr>
<tr>
<td>GT</td>
<td>Global temporary table</td>
<td>Displays the global temporary tables</td>
<td>SG, DB, TS</td>
</tr>
<tr>
<td>I</td>
<td>Insert</td>
<td>Adds a blank line to a list or inserts a step in a script</td>
<td>Work ID, baseline profile, migrate profile, script</td>
</tr>
<tr>
<td>ID</td>
<td>Identity column</td>
<td>In the Table Columns List panel, displays the Identity Column Detail panel, from which you can specify values for an identity column.</td>
<td>TB column</td>
</tr>
<tr>
<td>IX</td>
<td>Indexes</td>
<td>Displays the indexes of the specified object</td>
<td>SG, DB, TS, TB</td>
</tr>
<tr>
<td>KY</td>
<td>Keys</td>
<td>Displays the keys of an index, the partition keys of a table that uses table-controlled partitioning, or the constraint keys of a unique or primary constraint.</td>
<td>TBT, IX, UCP, UCU</td>
</tr>
<tr>
<td>L</td>
<td>Like</td>
<td>Creates a new data structure definition, an object, or an auxiliary object by using the current definition as a template.</td>
<td>SG, DB, TS, TS partition, TB, TBB, TB column, GT, CK, FK, IX, IX partition, VW, TR, AL, SY, SP, SQ</td>
</tr>
<tr>
<td>L</td>
<td>List</td>
<td>Lists baselines created from a baseline profile or task IDs created from an application or a script.</td>
<td>Baseline profile, application, script</td>
</tr>
<tr>
<td>Action code</td>
<td>Definition</td>
<td>Description</td>
<td>Use</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
<td>-------------</td>
<td>-----</td>
</tr>
<tr>
<td>LIX</td>
<td>List indexes</td>
<td>Displays a list of the auxiliary indexes that are associated with a large object (LOB) column in a base table (TBB). In the Table Columns List panel, this command can be used if the data type of the column is defined as a large object (LOB) (such as a binary large object, or BLOB; character large object, or CLOB; or double-byte character large object, or DBCLOB) or as a distinct or user-defined type (UDT) that is based on a LOB data type. You cannot use this command if you are liking or creating a TBB. You can, however, use it when you edit the TBB after it has been stored in the CD tables.</td>
<td>TBB column</td>
</tr>
<tr>
<td>LK</td>
<td>Limit key</td>
<td>In an index, table, or table space partition list, displays the limit key for a selected partition. In a mixed list, if the table uses table-controlled partitioning, you can type LK adjacent to the associated table space or table to display all of the limit keys for all of the partitions. You cannot type LK adjacent to an index in which the table uses table-controlled partitioning. In a mixed list, if the table uses index-controlled partitioning, you can type LK adjacent to the index to display all of the limit keys for all of the partitions. You cannot type LK adjacent to the associated table space or table.</td>
<td>TS, TST, TB, TBT, IX</td>
</tr>
<tr>
<td>Action code</td>
<td>Definition</td>
<td>Description</td>
<td>Use</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>LTB</td>
<td>List tables</td>
<td>Displays a list of the auxiliary tables that are associated with a large object (LOB) column in a base table (TBB). In the Table Columns List panel, this command can be used if the data type of the column is defined as a large object (LOB) (such as a binary large object, or BLOB; character large object, or CLOB; or double-byte character large object, or DBCLOB) or as a distinct or user-defined type (UDT) that is based on a LOB data type. This command cannot be used if you are liking or creating a TBB. It can, however, be used when you edit the TBB after it has been stored in the CD tables.</td>
<td>TBB column</td>
</tr>
<tr>
<td>LTS</td>
<td>List table spaces</td>
<td>Displays a list of the auxiliary table spaces that are associated with a large object (LOB) column in a base table (TBB). In the Table Columns List panel, this command can be used if the data type of the column is defined as a large object (LOB) (such as a binary large object, or BLOB; character large object, or CLOB; or double-byte character large object, or DBCLOB) or as a distinct or user-defined type (UDT) that is based on a LOB data type. This command cannot be used if you are liking or creating a TBB. It can, however, be used when you edit the TBB after it has been stored in the CD tables.</td>
<td>TBB column</td>
</tr>
<tr>
<td>M</td>
<td>Maintain change rules</td>
<td>Displays change rules for a location</td>
<td>Migrate profile</td>
</tr>
<tr>
<td>M</td>
<td>Migrate</td>
<td>Requests that the specified object be migrated. This action code is valid for a migrate-type work ID only. In the Mixed List panel, you cannot use the M action code on orphaned auxiliary objects.</td>
<td>SG, DB, TS, TB, GT, CK, FK, IX, VW, TR, AL, SY, SPE, SPS, SPN, SPNA, SQ</td>
</tr>
<tr>
<td>M</td>
<td>Move</td>
<td>Moves a key or column to the column before or after a particular column</td>
<td>TS, TB, FK, IX, UC, CK, TR, VW</td>
</tr>
<tr>
<td>MM</td>
<td>Block move</td>
<td>Moves a block of keys or columns from a list of columns</td>
<td>TS, TB, FK, IX, UC, CK, TR, VW</td>
</tr>
<tr>
<td>Action code</td>
<td>Definition</td>
<td>Description</td>
<td>Use</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>MO</td>
<td>Migrate options</td>
<td>Modifies the Migrate Options flag and marks the specified object for migration. This action code is valid for a migrate-type work ID only. To migrate the auxiliary objects for a large object (LOB) column, you must migrate the base table. In the Mixed List panel, you cannot use the MO action code on orphaned auxiliary objects.</td>
<td>SG, DB, TS, TB, TBB, GT, CK, FK, IX, VW, TR, AL, SY, SPE, SPS, SPN, SPNA, SQ</td>
</tr>
<tr>
<td>MT</td>
<td>Materialized query table (MQT) text</td>
<td>Displays the text associated with a materialized query table (MQT)</td>
<td>TBQT, TBQ</td>
</tr>
<tr>
<td>O</td>
<td>Options</td>
<td>Displays the options for a step in a script</td>
<td>Script</td>
</tr>
<tr>
<td>P</td>
<td>Protect</td>
<td>Indicates to protect a baseline from deletion</td>
<td>Baseline</td>
</tr>
<tr>
<td>PA</td>
<td>Propagate attributes</td>
<td>Propagates the values for the number of rows, primary quantity, secondary quantity, and allocation unit attributes of a selected partition to all of the partitions in an index or a table space</td>
<td>TS, IX</td>
</tr>
<tr>
<td>PE</td>
<td>Propagate estimates</td>
<td>Propagates the estimated space to the primary quantity for the specified partition in an index or a partitioned table space</td>
<td>TS, IX</td>
</tr>
<tr>
<td>PK</td>
<td>Primary key</td>
<td>Displays a list of columns that can be used in a unique or primary constraint</td>
<td>TB</td>
</tr>
<tr>
<td>PM</td>
<td>Procedure parameters</td>
<td>Displays a list of parameters for a stored procedure</td>
<td>SPE, SPN, SPNA</td>
</tr>
<tr>
<td>PO</td>
<td>Procedure options</td>
<td>Displays options for external stored procedures and native SQL stored procedures</td>
<td>SPE, SPN, SPNA</td>
</tr>
<tr>
<td>Action code</td>
<td>Definition</td>
<td>Description</td>
<td>Use</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
<td>-------------</td>
<td>-----</td>
</tr>
<tr>
<td>PT</td>
<td>Partitions</td>
<td>Displays information about the partitions of a table space or index. In a mixed list, if a table uses table-controlled partitioning, you can type PT adjacent to a TST or TBT to display information for all of the table space partitions. You cannot type PT adjacent to a table that uses index-controlled partitioning. This code also displays information for a nonpartitioned table space or index. For large (larger than 2 GB), nonpartitioned, VCAT-defined table spaces and indexes that are supported by multiple data sets, a row is displayed for each data set.</td>
<td>TS, TST, TBT, IX</td>
</tr>
<tr>
<td>QE</td>
<td>Quick edit</td>
<td>Marks tables for renaming Used in conjunction with the USE action code.</td>
<td>TB</td>
</tr>
<tr>
<td>R</td>
<td>Delete</td>
<td>Deletes baselines using the DELETEAGE and RETAINMAX values in the baseline profile</td>
<td>Baseline profile</td>
</tr>
<tr>
<td>R</td>
<td>Repeat</td>
<td>Creates an object from a template (synonym for Like action) Use a number in conjunction with this action code to apply the action multiple times.</td>
<td>SG, DB, TS, TS partition, TB, GT, CK, FK, IX, IX partition, VW, TR, AL, SY</td>
</tr>
<tr>
<td>R</td>
<td>Report</td>
<td>Selects a baseline for a baseline report</td>
<td>Baseline</td>
</tr>
<tr>
<td>RR</td>
<td>Block repeat</td>
<td>Repeats a block of keys or columns from a list of columns</td>
<td>TS, TB, FK, IX, UC, CK, TR, VW</td>
</tr>
<tr>
<td>S</td>
<td>Select</td>
<td>Selects one or more items from a list</td>
<td>Work ID, baseline, baseline profile, migrate profile, script, task ID, application</td>
</tr>
<tr>
<td>S</td>
<td>Specify</td>
<td>Invokes Specification for a work ID</td>
<td>Work ID</td>
</tr>
<tr>
<td>S</td>
<td>Steps</td>
<td>Displays the steps in a script</td>
<td>Script</td>
</tr>
<tr>
<td>SY</td>
<td>Synonym</td>
<td>Displays the synonyms of a table or view This command can be used from the BMC Environment panel for a plan.</td>
<td>SG, DB, TS, TB, GT, VW, PL</td>
</tr>
<tr>
<td>TB</td>
<td>Tables</td>
<td>Displays the dependent tables of the specified object</td>
<td>SG, DB, TS</td>
</tr>
<tr>
<td>Action code</td>
<td>Definition</td>
<td>Description</td>
<td>Use</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
<td>-------------</td>
<td>-----</td>
</tr>
<tr>
<td>TR</td>
<td>Trigger</td>
<td>Displays the trigger for a table</td>
<td>SG, DB, TS, TB</td>
</tr>
<tr>
<td>TS</td>
<td>Table spaces</td>
<td>Displays the dependent table spaces of the specified object</td>
<td>SG, DB</td>
</tr>
<tr>
<td>TX</td>
<td>Text</td>
<td>Displays the text associated with a check constraint, trigger, view, materialized query table (MQT), or XML index in the Mixed List panel</td>
<td>CK, IXV, IXVU, SP, TBQ, TBQT, TR, VW</td>
</tr>
<tr>
<td>TX</td>
<td>Text</td>
<td>Displays the text associated with an index on an expression in the Index Keys List or Index Key Mixed List panels</td>
<td>IXS, IXSU, IXST</td>
</tr>
<tr>
<td>U</td>
<td>Undo</td>
<td>Removes requested changes for the object or auxiliary object</td>
<td>SG, DB, TS, TSA, TB, TBB, GT, CK, FK, IX, IXA, VW, TR, AL, SY, SPE, SPS, SPN, SPNA, UC, SQ</td>
</tr>
<tr>
<td>U</td>
<td>Unprotect</td>
<td>Removes the protected designation from a baseline, thus allowing it to be available for deletion</td>
<td>Baseline</td>
</tr>
<tr>
<td>UC</td>
<td>Unique constraint zoom</td>
<td>Displays all of the unique or primary constraints for the specified object</td>
<td>DB, TS, TB, IX</td>
</tr>
<tr>
<td>USE</td>
<td>Use</td>
<td>Specifies the new table space for tables renamed using QE (quick edit)</td>
<td>TS</td>
</tr>
<tr>
<td>V</td>
<td>Execution status</td>
<td>Displays the execution status of a work ID</td>
<td>Work ID</td>
</tr>
<tr>
<td>V</td>
<td>View sync table</td>
<td>Displays the execution status of a task ID</td>
<td>Task ID</td>
</tr>
<tr>
<td>VL</td>
<td>Volume list</td>
<td>Displays a volume list from a Tablespace Parts List or Index Parts List panel</td>
<td>SG</td>
</tr>
<tr>
<td>VT</td>
<td>View text</td>
<td>Displays the text associated with a view</td>
<td>VW</td>
</tr>
<tr>
<td>VW</td>
<td>Views</td>
<td>Displays the dependent views of the specified object</td>
<td>SG, DB, TS, TB, GT, VW</td>
</tr>
<tr>
<td>W</td>
<td>Edit worklist</td>
<td>Edits the worklist for the specified work ID</td>
<td>Work ID</td>
</tr>
<tr>
<td>X</td>
<td>Create</td>
<td>Creates a baseline from a baseline profile or creates a task ID from an application or a script</td>
<td>Baseline profile, application, script</td>
</tr>
</tbody>
</table>
### Selection List action codes

You can perform the following actions directly from a Selection List panel without returning to the Action Menu panel and selecting an action.

#### Table 4: WORKID Selection List panel

<table>
<thead>
<tr>
<th>Action code</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>Edits the worklist</td>
</tr>
<tr>
<td>G</td>
<td>Generates CDL from the work ID</td>
</tr>
<tr>
<td>E</td>
<td>Edits the work ID</td>
</tr>
<tr>
<td>B</td>
<td>Browses the work ID</td>
</tr>
<tr>
<td>D</td>
<td>Deletes the work ID</td>
</tr>
<tr>
<td>S</td>
<td>Specifies the object types included in the work ID</td>
</tr>
<tr>
<td>A</td>
<td>Analyzes the work ID</td>
</tr>
<tr>
<td>X</td>
<td>Executes the work ID</td>
</tr>
<tr>
<td>V</td>
<td>Views execution status of the work ID</td>
</tr>
</tbody>
</table>

#### Table 5: Baseline Selection list panel

<table>
<thead>
<tr>
<th>Action code</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>Edits the comment</td>
</tr>
<tr>
<td>B</td>
<td>Browses the baseline</td>
</tr>
<tr>
<td>D</td>
<td>Deletes the baseline</td>
</tr>
<tr>
<td>P</td>
<td>Adds protection to the baseline</td>
</tr>
<tr>
<td>U</td>
<td>Removes protection from the baseline</td>
</tr>
</tbody>
</table>
### Change rules used with object attributes

You can use different types of change rules or change flag keywords to compare, import, or migrate attributes of the DB2 and application objects.

The change rules and change flag keywords are listed in Table 9 on page 47 and Table 10 on page 49. In these tables, abbreviations are used to indicate the types of change rules. Table 8 on page 45 lists the indicators that are used.

#### Table 8: Change rule indicators

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Use a change-type change rule to compare, import, or migrate.</td>
</tr>
<tr>
<td>F</td>
<td>Use a force-type change rule to migrate.</td>
</tr>
<tr>
<td>I</td>
<td>Use an include-type change rule to compare, import, or migrate.</td>
</tr>
<tr>
<td>Indicator</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>E</td>
<td>Use an exclude-type change rule to compare, import, or migrate.</td>
</tr>
<tr>
<td>S</td>
<td>Use a suppress-type change rule to compare or import.</td>
</tr>
<tr>
<td>P</td>
<td>Use the NOPARTITION change flag keyword to exclude from the comparison.</td>
</tr>
<tr>
<td>N</td>
<td>Use the following NOATTRIBUTE change flag keywords to exclude objects and attributes from the comparison:</td>
</tr>
<tr>
<td></td>
<td>■ NOBUFFERPOOL (for databases, table spaces, and indexes)</td>
</tr>
<tr>
<td></td>
<td>■ NOCOMPRESS</td>
</tr>
<tr>
<td></td>
<td>■ NODATABASE (object and attribute)</td>
</tr>
<tr>
<td></td>
<td>■ NODEFINE</td>
</tr>
<tr>
<td></td>
<td>■ NODROPRESTRICT</td>
</tr>
<tr>
<td></td>
<td>■ NOFREEPAGE</td>
</tr>
<tr>
<td></td>
<td>■ NOIDENTITY</td>
</tr>
<tr>
<td></td>
<td>■ NOLOCKSIZE</td>
</tr>
<tr>
<td></td>
<td>■ NOPARTITION</td>
</tr>
<tr>
<td></td>
<td>■ NOPATH</td>
</tr>
<tr>
<td></td>
<td>■ NOPCTFREE</td>
</tr>
<tr>
<td></td>
<td>■ NOPRIQTY</td>
</tr>
<tr>
<td></td>
<td>■ NOREMARKS</td>
</tr>
<tr>
<td></td>
<td>■ NOSECQTY</td>
</tr>
<tr>
<td></td>
<td>■ NOSEGSIZE</td>
</tr>
<tr>
<td></td>
<td>■ NOSTARTVALUE</td>
</tr>
<tr>
<td></td>
<td>■ NOSTOGROUP (object and attribute)</td>
</tr>
<tr>
<td></td>
<td>■ NOTABLESPACE (object and attribute)</td>
</tr>
<tr>
<td></td>
<td>■ NOTRACKMOD</td>
</tr>
<tr>
<td></td>
<td>■ NOVIEWCHECK</td>
</tr>
</tbody>
</table>
For more information about using change rules or change flag keywords in a comparison, refer to the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

Table 9 on page 47 lists the change rules and change flag keywords for DB2 objects.

### Table 9: Change rules and change flag keywords for DB2 objects

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SG</th>
<th>DB</th>
<th>TS</th>
<th>TB</th>
<th>CK</th>
<th>FK</th>
<th>IX</th>
<th>UC</th>
<th>SY</th>
<th>VW</th>
<th>TR</th>
<th>AL</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BUFFPOOLIX</td>
<td>C, S, N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BUFFPOOLTS</td>
<td>C, S, N</td>
<td>C, S, N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHECK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>CLOSE</td>
<td>C, S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COLNAME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPRESS</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CREATEDBY</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>CREATOR</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>DATABASE</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>DATACAPTURE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFINE</td>
<td>C, S, N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S, N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEVTYPE</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DROPRESTRICT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>DSSIZE</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EDITPROC</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERASE</td>
<td>C, S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>FIELDPROC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FREEPAGE</td>
<td>C, N, P, S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, N, P, S</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GBPCACHE</td>
<td>P</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRANTEE</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Chapter 1  Object types, commands, and action codes  47
<table>
<thead>
<tr>
<th>Attribute</th>
<th>SG</th>
<th>DB</th>
<th>TS</th>
<th>TB</th>
<th>CK</th>
<th>FK</th>
<th>IX</th>
<th>UC</th>
<th>SY</th>
<th>VW</th>
<th>TR</th>
<th>AL</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRANTOR</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>IDENTITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>INLINE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>LOCATION</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>LOCKMAX</td>
<td>C</td>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOCKSIZE</td>
<td>C</td>
<td>N</td>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>MAXPARTS</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAME</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>NUMPARTS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P</td>
</tr>
<tr>
<td>OWNER</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>PATH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>PCTFREE</td>
<td>C</td>
<td>E</td>
<td>N, P, S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, N, P, S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIECESIZE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>PRIQTY</td>
<td>C</td>
<td>N</td>
<td>P, S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, N, P, S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REMARKS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>SCHEMA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>SECQTY</td>
<td>C</td>
<td>N</td>
<td>P, S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, N, P, S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SEGSIZE</td>
<td>C</td>
<td>E</td>
<td>N, S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, N, P, S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STARTVALUE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOG_VCAT</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>STOGROUP</td>
<td>C</td>
<td>S</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, P, S, N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TABLESPACE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRACKMOD</td>
<td>C</td>
<td>E</td>
<td>N, P</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, N, P, S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VALIDPROC</td>
<td>C</td>
<td>d</td>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VALUES (LIMITKEY)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P</td>
</tr>
<tr>
<td>VCAT</td>
<td>C</td>
<td>S</td>
<td>C, F, S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, F, S</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
You can specify a change rule for a LOB table space by using the table space (TS) object.

You can specify a change rule for a LOB base table or an auxiliary table by using the table (TB) object.

You can specify a change rule for an auxiliary index by using the index (IX) object.

The product ignores this attribute for a change rule for an auxiliary table.

The product ignores this attribute for a change rule for a LOB table space.

The product treats this attribute as a comment unless the AUTHSW keyword or installation option is set to B.

The CREATOR and OWNER keywords correspond to the CREATOR column in the DB2 catalog tables for the objects.

The product applies this change rule only to table spaces with a DSSIZE greater than 0.

You cannot specify a change rule for a FIELDPROC attribute on a BIGINT, BINARY, or VARBINARY column in a table.

(DB2 Version 10 and later) The product applies this change rule only to an inline LOB column; the column must be in a table that is in a partition-by-growth or range-partitioned table space.

If you set the table space LOCKSIZE attribute to TABLESPACE or TABLE, you must set LOCKMAX to 0.

The product applies this change rule only to existing partition-by-growth table spaces.

You can include or exclude volumes from storage groups, VCAT-defined table spaces, or VCAT-defined indexes.

Table 10 on page 49 shows the types of change rules or change flag keywords that you can use to compare, import, or migrate attributes of the application objects. The All column indicates that a rule can be used for an attribute for all objects that are listed for the attribute.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SP</th>
<th>PG</th>
<th>PL</th>
<th>SQ</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVATE</td>
<td>C</td>
<td>a</td>
<td>b</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASUTIME</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUDIT</td>
<td>C</td>
<td></td>
<td></td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>BUFFPOOLIX</td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
<td></td>
</tr>
</tbody>
</table>

Table 10: Change rules and change flag keywords for application objects
<table>
<thead>
<tr>
<th>Attribute</th>
<th>SP</th>
<th>PG</th>
<th>PL</th>
<th>SQ</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFFPOOLTS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>CLOSE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>COLLECTION</td>
<td>C</td>
<td>C</td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>COLNAME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>COMPRESS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>CREATEDBY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>CREATOR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>DATABASE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>DATACAPTURE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>DBRMLIB</td>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>DEBUG_MODE</td>
<td>C</td>
<td>C</td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>DEFINE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>DEVTYPE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>EDITPROC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>ERASE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>FIELDPROC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>FREEPAGE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>GRANTEE</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>GRANTOR</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>LOCATION</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>LOCKMAX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>LOCKSIZE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>NAME</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>OWNER</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>PCTFREE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>PIECESIZE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>PRIQTY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
<tr>
<td>QUALIFIER</td>
<td></td>
<td>C</td>
<td>C</td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>RESIDENT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>SCHEMA</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>SEQQTY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>SEGSIZE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C, S</td>
</tr>
</tbody>
</table>

Change rules used with object attributes

50  ALTER and CHANGE MANAGER for DB2 Reference Manual
<table>
<thead>
<tr>
<th>Attribute</th>
<th>SP</th>
<th>PG</th>
<th>PL</th>
<th>SQ</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPALTER</td>
<td>C&lt;sub&gt;a&lt;/sub&gt; h&lt;sub&gt;b&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPTYPE</td>
<td>C&lt;sub&gt;a&lt;/sub&gt; i&lt;sub&gt;c&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOG_VCAT</td>
<td>C&lt;sub&gt;d&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>C&lt;sub&gt;e&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRACKMOD</td>
<td>C&lt;sub&gt;f&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VALIDPROC</td>
<td>C, S&lt;sub&gt;g&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VCAT</td>
<td>C&lt;sub&gt;h&lt;/sub&gt;, F, S&lt;sub&gt;i&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VCAT_STOG</td>
<td>C&lt;sub&gt;j&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VERSION</td>
<td>C&lt;sub&gt;k&lt;/sub&gt;, j&lt;sub&gt;l&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VOLORDER</td>
<td>C&lt;sub&gt;m&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VOLUME</td>
<td>C, E, I, S&lt;sub&gt;n&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WLM_ENV</td>
<td>C&lt;sub&gt;o&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

a. The Import and Compare components ignore this change rule.
b. To indicate the active version of a native SQL stored procedure, specify ALTER ACTIVATE VERSION in the DDL or CDL.
c. The product does not apply change rules for COLLECTION to trigger packages.
d. The product applies this change rule to native stored procedures and external stored procedures that are written in Java. Valid values for the current attribute are *, ALLOW, DISALLOW, or DISABLE. Valid values for the new attribute are ALLOW, DISALLOW, or DISABLE.
e. The NOLOCKSIZE keyword suppresses the LOCKMAX and the LOCKSIZE attributes.
f. The OWNER keyword corresponds to the OWNER column in the DB2 catalog tables for the objects.
g. The QUALIFIER keyword corresponds to the QUALIFIER column in the DB2 catalog tables for the objects.
h. The Import and Compare components will not change the CREATE PROCEDURE statement to an ALTER PROCEDURE statement.
i. The Import and Compare components will not convert an external SQL stored procedure to a native SQL stored procedure.
j. The product applies this change rule only to native SQL stored procedures.
k. Valid values for the current attribute are *, n*, or nnn. Valid values for the new attribute are *nnn or nnn.
l. Valid values for the new attribute are alphanumeric names or the <DEFLT> keyword. You can no longer specify <BLANK> or NO WLM for the new attribute value.
Keywords and variables

Several option values control the operating default environment. These values are defined in the installation options module, an assembler module that the components of the ALTER and CHANGE MANAGER products use. In addition to using this module, each component that can run in batch mode reads an input stream that contains parameters and option override values for the component.

Keywords for input streams

The following table lists the input streams. Subsequent tables describe the keywords that are associated with each input stream.

Table 11: Input streams

<table>
<thead>
<tr>
<th>Input stream</th>
<th>Component</th>
<th>Table of keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>AEXIN</td>
<td>Execution</td>
<td>“AEXIN keywords” on page 55</td>
</tr>
<tr>
<td>AEXPIN</td>
<td>Execution for the worklist parallelism feature of the Database Administration and BMC Object Administration for DB2 solutions AEXPIN is not generated if the -BEGG worklist parallelism command is not the first command in the worklist.</td>
<td>“AEXPIN keywords” on page 62</td>
</tr>
<tr>
<td>AJXIN</td>
<td>Batch Execution JCL Generation</td>
<td>AJXIN keywords on page 64</td>
</tr>
<tr>
<td>AJXPOFIN</td>
<td>Batch Execution JCL Generation Batch Analysis JCL Generation The AJXPOFIN input stream includes most of the keywords that the product options file (POF) includes.</td>
<td>“JCL Generation product options” on page 181</td>
</tr>
<tr>
<td>ALUIN</td>
<td>Other components</td>
<td>ALUIN keywords on page 71</td>
</tr>
</tbody>
</table>
Overview of the keyword tables

Each keyword table provides a description of the keyword and the default.

The tables provide the following information:

- **Keyword**
  This column lists the keyword that specifies the option in the input stream of one or more of the components. Most keywords do not require a parameter. If a parameter is required, an appropriate placeholder in lowercase letters follows the keyword.
  
  The placeholder is described in the **Description** column. For example, HSMVOL `volumeID` denotes that the HSMVOL keyword must be followed by a volume ID. The placeholder `a.b` represents a two-part identifier, such as that used for work IDs, profiles, and baselines.

- **Default**
  For the POF keywords, this column lists the default value for the keyword. Some keywords do not have a default value. In that case, the **Default** column specifies **none**.

- **Corresponding installation option**
  This column lists the corresponding parameter that is used in the installation options module. In the installation options module, the parameter is followed by an equal sign (=) and a value. If the option lists more than one value, the values are separated by commas and enclosed within square brackets. For example, `AUTHSW=[Y, N, B, X]` means that the AUTHSW parameter can have a value of Y, N, B, or X.

  **Note**
  Some keywords do not have corresponding installation options. In that case, the **Corresponding installation option** column specifies **None**, or the table omits the column.

- **Description**
  This column describes the purpose of the keyword and sometimes includes usage notes.

- **Used by**
  This column specifies which components use the specified keyword. Table 12 on page 55 lists each component and the abbreviation for each component.
Table 12: Components

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Analysis</td>
</tr>
<tr>
<td>B</td>
<td>Baseline</td>
</tr>
<tr>
<td>BR</td>
<td>Baseline Report</td>
</tr>
<tr>
<td>C</td>
<td>Compare</td>
</tr>
<tr>
<td>E</td>
<td>Execution</td>
</tr>
<tr>
<td>F</td>
<td>Front End or JCL Generation</td>
</tr>
<tr>
<td>I</td>
<td>Import</td>
</tr>
<tr>
<td>J</td>
<td>Batch Execution JCL Generation</td>
</tr>
<tr>
<td>P</td>
<td>CM/PILOT</td>
</tr>
<tr>
<td>S</td>
<td>Specification</td>
</tr>
</tbody>
</table>

For a complete listing of all installation options, see “ALTER and CHANGE MANAGER installation options” on page 159.

For instructions about setting installation option values at installation, see:

- *Installation System Reference Manual*
- *Installation System Quick Start*
- *BMC Products and Solutions for DB2 Customization Guide*

### AEXIN keywords

The following table lists the keywords in the AEXIN input stream.

Table 13: AEXIN keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>2MEGSQ</td>
<td>Instructs the Execution component to allocate a 2-MB buffer for large SQL statements</td>
<td>E, F, J</td>
</tr>
<tr>
<td>ACM</td>
<td>Specifies the CHANGE MANAGER product</td>
<td>E</td>
</tr>
<tr>
<td>ALTERID</td>
<td>Specifies the name of the ALTER ID</td>
<td>E</td>
</tr>
<tr>
<td>ALU</td>
<td>Specifies the ALTER product</td>
<td>E</td>
</tr>
<tr>
<td><strong>Keyword</strong></td>
<td><strong>Description</strong></td>
<td><strong>Used by</strong></td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>ASU</strong></td>
<td>Specifies the DASD MANAGER PLUS product</td>
<td>E</td>
</tr>
<tr>
<td><strong>AUC</strong></td>
<td>Specifies the CM/PILOT component of the CHANGE MANAGER product</td>
<td>E</td>
</tr>
<tr>
<td><strong>BINDFAIL</strong></td>
<td>Causes worklist execution to stop with a return code of 8 if a bind fails. The halt will be noted in the sync tables, and an Execution restart will continue with the command that caused the failure. Without this parameter, worklist execution continues if a bind fails.</td>
<td>E, F, J</td>
</tr>
<tr>
<td><strong>CATAUDIT</strong></td>
<td>If the CATAUDIT installation option is set to N and you manually add the keyword to the AEXIN input stream, instructs Execution to perform the following tasks: - Invoke the CATALOG MANAGER product - Override the installation option - Log executed DDL statements in the CATALOG MANAGER DDL Audit Log If the keyword is not in the AEXIN input stream, Execution reads the ALTER or CHANGE MANAGER installation options. If CATAUDIT=Y, Execution audits the worklist. JCL Generation includes the CATALOG MANAGER installation option in the AEXIN input stream. The name of the installation option is passed from the BMCDB2 control table to CATALOG MANAGER when the product is invoked. This parameter is useful only if you have installed CATALOG MANAGER. You must use CATALOG MANAGER to recover any dropped objects.</td>
<td>E</td>
</tr>
<tr>
<td><strong>CATDOPT name</strong></td>
<td>Provides the ALTER or CHANGE MANAGER products with the name of the installation options module for the BMC CATALOG MANAGER product</td>
<td>E, F, J</td>
</tr>
<tr>
<td>Keyword</td>
<td>Description</td>
<td>Used by</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
</tbody>
</table>
| CATRECOVER     | If the CATRECOV installation option is set to N and the keyword is in the AEXIN input stream, instructs Execution to perform the following tasks:  
- Invoke the BMC CATALOG MANAGER product  
- Log the information that is required to recover any objects that are dropped in the worklist  
You can manually insert the keyword into the AEXIN input stream, or you can use the Drop Recovery option on the Execution Override Options panel.  
JCL Generation reads the ALTER or CHANGE MANAGER installation options. If CATRECOV=Y, JCL Generation inserts the keyword into the AEXIN input stream. If you set the CATRECOV installation option to N and rerun Execution, or if you manually remove the keyword from the AEXIN input stream, the dropped objects are not logged or rebuilt.  
JCL Generation includes the CATALOG MANAGER installation option in the AEXIN input stream. The name of the installation option is passed from the BMCDB2 control table to CATALOG MANAGER when the product is invoked.  
This parameter is useful only if you have installed CATALOG MANAGER. You must use CATALOG MANAGER to recover any dropped objects. | E, F    |
| CATUTIL        | For CATALOG MANAGER, specifies the worklist job                                                                                                                                                          | NA      |
| CHECKOPT       | Provides the BMC products with the name of the options module for the CHECK PLUS product  
The BMC_CHECK_OPTS keyword in the AJXPOFIN input stream replaces this keyword.                                                                 | E       |
| COPYOPT        | Provides the BMC products with the name of the options module for the NGT Copy product  
The BMC_COPY_OPTS keyword in the AJXPOFIN input stream replaces this keyword.                                                                 | E       |
| DASDDOPT name  | Provides ALTER or CHANGE MANAGER with the name of the installation options module for the DASD MANAGER PLUS product  
This keyword allows Execution to access the BMCSTATS utility.                                                                                   | E, F, J |
<p>| DASDTRIG       | For DASD MANAGER PLUS, specifies running a triggered job                                                                                                                                                  | NA      |</p>
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB2STMSGS</td>
<td>Instructs Execution to send to AEXPRINT all of the messages that are returned from DB2 when a DB2 STOP command is issued. Usually, these messages are suppressed, and the contents are analyzed by the STOPWAIT processor. However, if the stop does not occur, you might want to view the original messages.</td>
<td>E</td>
</tr>
<tr>
<td>DEBUGUNLD</td>
<td>Enables debugging user exits, if the exits are coded to use the keyword. This keyword also instructs Execution to turn on bit VAUNFDBG in flag field VAUNFLAG. Execution passes this bit to the user exit in the control block.</td>
<td>E</td>
</tr>
<tr>
<td>DYNWORKUNIT</td>
<td>For the Database Administration, BMC Next Generation Technology Database Administration for DB2, and BMC Object Administration for DB2 solutions, defines the unit (such as SYSDA) that Execution uses to dynamically allocate temporary work data sets.</td>
<td>E</td>
</tr>
<tr>
<td>ENV</td>
<td>Instructs the component to print ALTER or CHANGE MANAGER environment information (including a list of indexes Analysis uses) in its diagnostic output. The ALUIN input stream also uses this keyword.</td>
<td>A, B, BR, C, E, F, I, J</td>
</tr>
<tr>
<td>EVENTS</td>
<td>For DASD MANAGER PLUS, specifies whether to record utilities in an events table.</td>
<td>NA</td>
</tr>
<tr>
<td>FLOW</td>
<td>Causes Execution to produce flow trace messages in AEXPRINT when entering and exiting modules.</td>
<td>E</td>
</tr>
<tr>
<td>HASHFAIL</td>
<td>Causes Execution to terminate the job if a hash failure, such as a changed or added statement, occurs in a worklist.</td>
<td>E, F, J</td>
</tr>
<tr>
<td>HASHWARNRC returnCode</td>
<td>Defines the return code (returnCode) that Execution sends back when it finds only hash warnings. Do not use 8 for this value.</td>
<td>E, F, J</td>
</tr>
<tr>
<td>ITERATIONMODE</td>
<td>Enables Execution to run a REXX executable that generates utility syntax or SQL for Execution to run based on user-defined criteria. This utility syntax or SQL must be in a worklist format. You are responsible for creating the JCL that would enable the utility to run correctly (such as using dynamically allocated work files in the utilities). The REXX executable runs iteratively until it returns a code greater than 4 to Execution. You are responsible for restarting any commands that failed. Execution does not record any actions in the sync table. You must manually insert this keyword in the AEXIN input stream.</td>
<td>E</td>
</tr>
</tbody>
</table>

58  *ALTER and CHANGE MANAGER for DB2 Reference Manual*
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINES n</td>
<td>Provides Execution with the number (n) of output lines per print page for AEXPRINT. The default is 56 lines per page. If you specify this keyword, you might need to change or add a /*JOBPARAM statement in your JCL. You can manually add or modify this keyword.</td>
<td>E</td>
</tr>
<tr>
<td>LOADOPT</td>
<td>Provides the BMC products with the name of the options module for the LOADPLUS product. The BMC_LOAD_OPTS keyword in the AJXPOFIN input stream replaces this keyword.</td>
<td>E</td>
</tr>
<tr>
<td>NEWTASKID a.b</td>
<td>For CHANGE MANAGER, creates a new task ID.</td>
<td>E, P</td>
</tr>
<tr>
<td>NEWWORKID a.b</td>
<td>For CHANGE MANAGER, creates a new work ID. EXECUTION uses this keyword in batch mode only. The ALUIN input stream also uses this keyword.</td>
<td>E, I, P</td>
</tr>
<tr>
<td>NOAPFOK</td>
<td>Does not perform APF checking.</td>
<td>E</td>
</tr>
</tbody>
</table>
| NOFAILNOIMAGECPY | For ALTER, CHANGE MANAGER, BMC Workbench, and CATALOG MANAGER, instructs Execution to invoke the Drop Recovery feature of the CATALOG MANAGER product and to allow an object to be dropped when the following conditions exist:  
  - No image copies of the object exist.  
  - The CATRECOVER keyword is specified in the AEXIN input stream.  
The product automatically generates the NOFAILNOIMAGECPY keyword in the AEXIN input stream when one of the following conditions exists:  
  - The DROPR_NOIC POF keyword is set to Y.  
  - The NOFAILNOIMAGECPY option is set to Y on the JCL Generation Debugging, Display and Execution panel.  
This keyword is useful only if you have installed CATALOG MANAGER. This keyword overrides the DROPR_NOIC keyword in the AJXPOFIN input stream. | E, J                     |
<p>| NOLOADCOMP       | Instructs Execution not to compress extra spaces out of LOAD statements.                                                                                                                                          | E                        |
| NOSQLCOMP        | Instructs Execution not to compress extra spaces out of SQL statements.                                                                                                                                       | E                        |
| NOSTARTOVER      | Instructs Execution not to start the worklist again from the beginning of the worklist. See also the STARTOVER keyword in this table.                                                                         | E, P                     |</p>
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOTIFYUNLD $n$</td>
<td>Instructs Execution to send a message to AEXPRINT every $n$ records during an unload</td>
<td>E</td>
</tr>
<tr>
<td>NOWKIDREPLACE</td>
<td>For CHANGE MANAGER, instructs Import not to replace the changes in an existing work ID with an imported file</td>
<td>E, P</td>
</tr>
<tr>
<td>REBINDFAIL</td>
<td>Causes worklist execution to stop with return code 8 if a rebind fails The stop is noted in the sync tables, and an Execution restart continues with the command that caused the failure. Without this parameter, worklist execution continues if a rebind fails.</td>
<td>E, F, J</td>
</tr>
<tr>
<td>REBINDRC $n$</td>
<td>Allows worklist execution to continue if a rebind fails, but returns the $n$ value for a final condition code instead of 4, the default value for the final condition code When running standard JCL, the condition code is added to the step subsequent to the REBIND step. Execution writes warning messages to AEXPRINT but does not post entries in the sync tables.</td>
<td>E, F, J</td>
</tr>
<tr>
<td>RECOVEROPT</td>
<td>Provides the BMC products with the name of the options module for the NGT Recover product The BMC_RECOVER_OPTS keyword in the AJXPofin input stream replaces this keyword.</td>
<td>E</td>
</tr>
<tr>
<td>REORGOPT</td>
<td>Provides the BMC products with the name of the options module for the REORG PLUS product The BMC_REORG_OPTS keyword in the AJXPofin input stream replaces this keyword.</td>
<td>E</td>
</tr>
<tr>
<td>REPLACEASKID a.b</td>
<td>For CHANGE MANAGER, replaces an existing task ID</td>
<td>E, P</td>
</tr>
<tr>
<td>REPLACEWORKID a.b</td>
<td>For CHANGE MANAGER, replaces an existing work ID The ALUIN input stream also uses this keyword.</td>
<td>E, F, I, P</td>
</tr>
<tr>
<td>RESTART</td>
<td>Instructs Execution to restart a worklist from the last sync or stop point RESTART fails if no -STOP command or error sync point (-SYNC) exists in the worklist. You cannot specify the RESTART keyword with the STARTOVER keyword.</td>
<td>E, J</td>
</tr>
<tr>
<td>RESTARTPARM parameter</td>
<td>During Execution restart, passes a user-defined parameter string (parameter) to the utility that is being restarted The form of the parameter string depends on the utility that is being restarted.</td>
<td>E, J</td>
</tr>
<tr>
<td>SPBXPRINT</td>
<td>Prints the output from a stored procedure</td>
<td>E</td>
</tr>
<tr>
<td>Keyword</td>
<td>Description</td>
<td>Used by</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
</tbody>
</table>
| SSID ssid   | Identifies the DB2 subsystem ID or the DB2 data sharing group attachment name  
The SSID keyword must match the -SSID command in the worklist. This parameter is required.  
If the SSID is specified in the JCL in an EXEC statement in a cataloged procedure (which begins with a PROC statement), its value overrides the value of the SSID keyword.  
The AJXIN and ALUIN input streams also use this keyword. | A, B, BR, E, F, I, J, P, S |
| STARTOVER   | Instructs Execution to start the worklist again from the beginning of the worklist  
You cannot specify the STARTOVER keyword with the RESTART keyword. See also the NOSTARTOVER keyword in this table.                                                                                     | E, J, P                |
| STATS       | Prints the execution statistics                                                                                                                                                                             | E                      |
| STOPWAIT n  | Specifies the number (n) of intervals to wait for a DB2 STOP command to stop a database or table space  
The first interval is 10 seconds, and the second through tenth intervals are an additional 30 seconds each. Examples are as follows:  
- n = 1, total wait = 10 seconds  
- n = 2, total wait = 40 seconds  
- n = 3, total wait = 70 seconds  
- n = 4, total wait = 100 seconds  
- n = 5, total wait = 130 seconds  
- n = 6, total wait = 160 seconds  
- n = 7, total wait = 190 seconds  
- n = 8, total wait = 220 seconds  
- n = 9, total wait = 250 seconds  
- n = 10, total wait = 280 seconds  
If the Execution program encounters a DB2 STOP command, Execution checks the object status to see whether the object actually stopped. If not, the Execution program waits for the specified interval and checks again.  
If n number of intervals passes without the object stopping successfully, the Execution program terminates with a -STOP command. If such a stop occurs, you can restart Execution when the object finally stops.  
The default value is 3. A value of 0 indicates that if the object does not stop, the worklist stops without waiting. The maximum value allowed for this keyword is 10 (which is 280 seconds). | E, F, J               |
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Used by</th>
</tr>
</thead>
</table>
| STOPWTSECS n    | Specifies the number (n) of seconds to wait in the first interval of the STOPWAIT keyword for a DB2 STOP command to stop a database or table space  
By default the length of the first interval is 10 seconds. See also the STOPWAIT AEXIN keyword in this table. | E, F, J                |
| SYNDELETE       | Instructs Execution to remove all sync entries when an Execution job completes with no errors                                                                                                                  | E, F, J                |
| SYNLIST         | Prints a synonym list                                                                                                                                                                                      | E                      |
| TASKID a.b      | For CHANGE MANAGER, specifies the task ID to use                                                                                                                                                          | E, P                   |
| UNLOADDOPT      | Specifies the name of the options module for the UNLOAD PLUS product  
The BMC_UNLOAD_OPTS keyword in the AJXPOFIN input stream replaces this keyword.                                                                 | E                      |
| UTILITYID       | For DASD MANAGER PLUS, specifies the utility ID to use for the utilities                                                                                                                                   | NA                     |
| VCAT            | For DASD MANAGER PLUS, identifies the VCAT for jobs  
The AJXIN input stream also uses this keyword.                                                                                                                                                             | NA                     |
| WARNRC          | Specifies the return code to use for warnings                                                                                                                                                              | E                      |
| WORKID a. b     | Specifies the work ID to use  
Execution fails if this work ID does not match the work ID that the -WKID command in the worklist specifies. The AJXIN and ALUIN input streams also use this keyword. | A, B, E, I             |

**AEXPIN keywords**

The AEXPIN input stream is used only by the Database Administration and BMC Object Administration for DB2 solutions, which use AEXPIN for worklist parallelism.

This section lists the keywords in the AEXPIN input stream. For the list of components to which the Used by column refers, see “AEXIN keywords” on page 55.
### Table 14: AEXPIN keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Default</th>
<th>Description</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXINITS</td>
<td>3</td>
<td>Specifies the maximum number of initiators for the BMC Cross-System Image Manager (XIM) technology. This keyword overrides the ACM_PARALLEL_MAXINIT keyword in the AJXPOFIN input stream.</td>
<td>E, F</td>
</tr>
<tr>
<td>MININITS</td>
<td>2</td>
<td>Specifies the minimum number of initiators for the BMC Cross-System Image Manager (XIM) technology. This keyword overrides the ACM_PARALLEL_MININIT keyword in the AJXPOFIN input stream.</td>
<td>E, F</td>
</tr>
<tr>
<td>PARALLEL</td>
<td>N</td>
<td>Specifies whether to run a worklist in parallel. This keyword overrides the ACM_PARALLEL_WORKLST keyword in the AJXPOFIN input stream.</td>
<td>E, F</td>
</tr>
<tr>
<td>STATUSINFO</td>
<td>NONE</td>
<td>Specifies not to collect information about the status of a unit of work (UOW) that is executing in a BMC Cross-System Image Manager (XIM) initiator. You must manually insert this keyword in the AEXPIN input stream.</td>
<td>E</td>
</tr>
<tr>
<td>TRACE</td>
<td>N</td>
<td>Specifies whether to display trace messages in a tracing output log (AEXPTRAC) while executing a worklist that is running in parallel. This keyword overrides the ACM_PARALLEL_XIMTRCE keyword in the AJXPOFIN input stream.</td>
<td>E, F</td>
</tr>
<tr>
<td>XIMGROUP</td>
<td>None</td>
<td>Specifies the group name for the BMC Cross-System Image Manager (XIM) technology. The group name for XIM must be unique for each instance of XIM that is running on an OS/390 or z/OS image. This keyword overrides the ACM_PARALLEL_XIMGRP keyword in the AJXPOFIN input stream.</td>
<td>E, F</td>
</tr>
<tr>
<td>XIMPCTSURMN</td>
<td>None</td>
<td>Specifies the percentage of the total computing resources that should be reserved. For example, if you set XIMPCTSURMN to 10 percent and the total service unit (SU) capacity of an image is 45,000 SU per second, a BMC Cross-System Image Manager (XIM) initiator will not start if the estimated SU availability is less than 4,500 SUs. You must manually insert this keyword in the AEXPIN input stream.</td>
<td>E</td>
</tr>
</tbody>
</table>
### Keywords for input streams

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Default</th>
<th>Description</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>XIMPROC</td>
<td>XIMACM</td>
<td>Specifies the name of the started procedure for the BMC Cross-System Image Manager (XIM) technology. BMC recommends that the name of the XIM started procedure be unique for each instance of XIM that is running on an OS/390 or z/OS image. This keyword overrides the ACM_PARALLEL_XIMPROC keyword in the AJXPOFIN input stream.</td>
<td>E, F</td>
</tr>
<tr>
<td>XIMRETRYCOUNT</td>
<td>None</td>
<td>Specifies the maximum number of attempts that Execution makes to schedule the initiators that could not be scheduled because of resource shortages. Execution terminates if it cannot allocate the initiators within the specified number of retry attempts. You must manually insert this keyword in the AEXPIN input stream.</td>
<td>E</td>
</tr>
<tr>
<td>XIMRETRYMINS</td>
<td>None</td>
<td>Specifies the number of minutes that Execution waits until it attempts to schedule initiators that could not be scheduled because of resource shortages. You cannot specify both the XIMRETRYMINS keyword and the XIMRETRYSECS keyword. You must manually insert this keyword in the AEXPIN input stream.</td>
<td>E</td>
</tr>
<tr>
<td>XIMRETRYSECS</td>
<td>None</td>
<td>Specifies the number of seconds that Execution waits until it attempts to schedule initiators that could not be scheduled because of resource shortages. You cannot specify both the XIMRETRYSECS keyword and the XIMRETRYMINS keyword. You must manually insert this keyword in the AEXPIN input stream.</td>
<td>E</td>
</tr>
<tr>
<td>XIMSTART</td>
<td>N</td>
<td>Specifies whether to start the BMC Cross-System Image Manager (XIM) technology automatically on the OS/390 or z/OS image on which the job is submitted. This keyword is applicable if XIM is not active when the worklist is executed. XIMSTART overrides the ACM_PARALLEL_XIMSTRT keyword in the AJXPOFIN input stream.</td>
<td>E, F</td>
</tr>
</tbody>
</table>

### AJXIN keywords

The following table lists the keywords in the AJXIN input stream. Unless otherwise noted in the description column, none of the AJXIN keywords has a corresponding installation option.
For the list of components to which the Used by CHANGE MANAGER component column refers, see AEXIN keywords on page 55.

Table 15: AJXIN keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>AJXJCLDDN</td>
<td>Specifies an alternate DD statement for AJXJCL</td>
<td>J</td>
</tr>
<tr>
<td>AMS</td>
<td>Specifies the commands for the IBM SMS</td>
<td>J</td>
</tr>
<tr>
<td>ANALYZE</td>
<td>For ALTER and CHANGE MANAGER, instructs Batch Execution JCL Generation to generate JCL for Analysis</td>
<td>J</td>
</tr>
<tr>
<td>ANLBASELINE baseline</td>
<td>For CHANGE MANAGER, specifies the name of a baseline for the JCL for Analysis</td>
<td>J</td>
</tr>
<tr>
<td>ANLBLPROFILE baselineProfile</td>
<td>For CHANGE MANAGER, specifies the name of the baseline profile for the JCL for Analysis</td>
<td>J</td>
</tr>
<tr>
<td>ANLDIAG fileName</td>
<td>For ALTER and CHANGE MANAGER, specifies the diagnostic output file for the JCL for Analysis</td>
<td>J</td>
</tr>
<tr>
<td>ANLMIGPROFILE migrateProfile</td>
<td>For CHANGE MANAGER, specifies the name of a migrate profile for the JCL for Analysis</td>
<td>J</td>
</tr>
<tr>
<td>ARCHSTAK</td>
<td>Specifies to stack archive discard data sets on tape</td>
<td>J</td>
</tr>
<tr>
<td>BEGJOBSEQ</td>
<td>For the BMCTRIG component of DASD MANAGER PLUS, specifies the beginning job sequence number</td>
<td>N/A</td>
</tr>
<tr>
<td>BIND</td>
<td>Specifies the commands for binding bound packages and plans</td>
<td>J</td>
</tr>
<tr>
<td>CATDOPT installationOptionsModule</td>
<td>Specifies the name of the installation options module for the CATALOG MANAGER product</td>
<td>E, F, J</td>
</tr>
<tr>
<td>CHECKDOPT installationOptionsModule</td>
<td>Specifies the name of the installation options module for the CHECK PLUS utility</td>
<td>F, J</td>
</tr>
<tr>
<td>COPY01 parameters</td>
<td>Specifies the local-copy parameters (parameters) for the COPYDDN keyword</td>
<td>J</td>
</tr>
<tr>
<td></td>
<td>The corresponding installation option is COPYDD01.</td>
<td></td>
</tr>
<tr>
<td>COPY02 parameters</td>
<td>Specifies the local-copy parameters (parameters) for the COPYDDN keyword</td>
<td>J</td>
</tr>
<tr>
<td></td>
<td>The corresponding installation option is COPYDD02.</td>
<td></td>
</tr>
<tr>
<td>Keyword</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>COPYDOPT</td>
<td>Specifies the name of the installation options module for the NGT Copy utility. The BMC_COPY_OPTS keyword in the AJXPOFIN input stream replaces this keyword.</td>
<td>F, J</td>
</tr>
<tr>
<td>DASDDOPT</td>
<td>Specifies the name of the installation options module for the DASD MANAGER PLUS product. This keyword allows Execution to access the BMCSTATS utility.</td>
<td>E, F, J</td>
</tr>
</tbody>
</table>
| DATASETSIZING [N,C,B,O] | Specifies the type of data set sizing. The following values are valid:  
- N does not perform data set sizing  
- C uses IBM RUNSTATS to perform data set sizing by using statistics from the DB2 catalog  
- B uses BMCSTATS to perform data set sizing by using the statistics from the DASD MANAGER PLUS product tables  
- O physically and randomly samples the VSAM objects to estimate data set sizes | J |
<p>| DBRM1 libraryName | Names three default application DBRM libraries that JCL Generation uses when generating Execution JCL when BIND PLAN statements are included in the worklist. The corresponding installation options are DBRM1, DBRM2, DBRM3. | E, J |
| DEFGDGBASE | Creates the base of the generation data group (GDG) | J |
| DEFGDGLIMIT | Specifies the maximum number of generation data group (GDG) data sets that are allowed | J |
| DELETEAGEnnnn | For CHANGE MANAGER, deletes baselines that are older than the current date. See also the DELETEAGE keyword in the ALUIN input stream. | J |
| DSNHLQ | For ALTER and CHANGE MANAGER, overrides the ZUSER symbolic variable | F |</p>
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>EURO</td>
<td>Instructs ALTER and CHANGE MANAGER to expect numbers in the European format (which uses a comma for the decimal point) and to create output in European decimal format. This parameter is particularly important when ALTER and CHANGE MANAGER parse index LIMITKEY values that are separated by commas. If the EURO keyword is present, ALTER and CHANGE MANAGER require delimiting commas to be followed by blanks. The corresponding installation option is EURO. The Import, Specification, Baseline, and Compare components use the value for EURO from the installation options module but do not support use of the EURO keyword in the ALUIN input data stream. The ALUIN input stream also uses this keyword.</td>
<td>A, B, BR, C, I, J, S</td>
</tr>
<tr>
<td>GLID globalAuthorizationID</td>
<td>Defines a global authorization ID (GLID)</td>
<td>J</td>
</tr>
<tr>
<td>IMPDIAG fileName</td>
<td>For ALTER and CHANGE MANAGER, specifies the diagnostic output file for the JCL for Import</td>
<td>J</td>
</tr>
<tr>
<td>IMPMIGPROFILE migrateProfile</td>
<td>For CHANGE MANAGER, specifies the name of a migrate profile for the JCL for Import</td>
<td>J</td>
</tr>
<tr>
<td>JCLDEBUG Y</td>
<td>Generates debugging comments in the generated JCL</td>
<td>F, J</td>
</tr>
<tr>
<td>JOBACPS</td>
<td>Defines the primary space allocation for the archive data set</td>
<td>J</td>
</tr>
<tr>
<td>JOBACSS</td>
<td>Defines the secondary space allocation for the archive data set</td>
<td>J</td>
</tr>
<tr>
<td>JOBSEQTYP</td>
<td>For the BMCTRIG component of DASD MANAGER PLUS, specifies the job sequence type</td>
<td>J</td>
</tr>
<tr>
<td>JULIANDATE</td>
<td>Specifies the date that the work ID was created</td>
<td>J</td>
</tr>
</tbody>
</table>
| LOADDOPT installationOptionsModule | Specifies the name of the installation options module for the LOADPLUS utility  
The BMC_LOAD_OPTS keyword in the AJXPOFIN input stream replaces this keyword.                                                                 | F, J                             |
<p>| NOTIFY tsoID            | Specifies the TSO ID to notify when a job is submitted                                                                                                                                                   | J                                |
| POFDS dataSetName member | For ALTER, CATALOG MANAGER, and CHANGE MANAGER, identifies the data set and member name for the product options file (POF)                                                                              | F, J                             |
| POFDSN                  | For DASD MANAGER PLUS, identifies the data set and member name for the product options file (POF)                                                                                                          | J                                |</p>
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>POSTBASELINE baseline</td>
<td>For CHANGE MANAGER, creates a post-Execution baseline</td>
<td>E, J</td>
</tr>
<tr>
<td>POSTBDIAG [SYSOUT, dataSetName]</td>
<td>Specifies a location for a post-Execution baseline diagnostic file</td>
<td>E, J</td>
</tr>
<tr>
<td>POSTBNAME baseline</td>
<td>For CHANGE MANAGER, specifies a baseline name for a post-Execution baseline</td>
<td>E, J</td>
</tr>
<tr>
<td>POSTBPROFILE baselineProfile</td>
<td>For CHANGE MANAGER, specifies a profile name for a post-Execution baseline</td>
<td>E, J</td>
</tr>
<tr>
<td>POSTBPROFILE baselineProfile</td>
<td>Specifies a baseline name for a post-Execution comparison</td>
<td>E, J</td>
</tr>
<tr>
<td>POSTCCDL dataSetName</td>
<td>For CHANGE MANAGER, specifies the data set name for the Change Definition Language (CDL) from a post-Execution step</td>
<td>J</td>
</tr>
<tr>
<td>POSTCDIAG [SYSOUT, dataSetName]</td>
<td>For CHANGE MANAGER, specifies a location for a post-Execution comparison diagnostic file</td>
<td>E, J</td>
</tr>
<tr>
<td>POSTCDL dataSetName</td>
<td>For CHANGE MANAGER, specifies the data set name for the CDL from a post-Execution comparison</td>
<td>E, J</td>
</tr>
<tr>
<td>POSTCOMPARE [C,O]</td>
<td>For CHANGE MANAGER, specifies a post-Execution comparison The following values are valid: ■ C specifies a previous baseline-to-catalog comparison. ■ O specifies the comparison of a baseline built in a previous job step to the catalog.</td>
<td>E, J</td>
</tr>
<tr>
<td>PREBASELINE baseline</td>
<td>For CHANGE MANAGER, creates a pre-Execution baseline</td>
<td>E, J</td>
</tr>
<tr>
<td>PREBDIAG [SYSOUT, dataSetName]</td>
<td>For CHANGE MANAGER, specifies a location for a pre-Execution baseline diagnostic file</td>
<td>E, J</td>
</tr>
<tr>
<td>PREBNAME baseline</td>
<td>For CHANGE MANAGER, specifies a baseline name for a pre-Execution baseline</td>
<td>E, J</td>
</tr>
<tr>
<td>PREBPROFILE baselineProfile</td>
<td>For CHANGE MANAGER, specifies a profile name for a pre-Execution baseline</td>
<td>E, J</td>
</tr>
<tr>
<td>PRECBASE baseline1</td>
<td>For CHANGE MANAGER, specifies, to the catalog, either the name of baseline 1 that the components use in a pre-Execution comparison of previously created baselines, or the baseline name for a pre-Execution baseline</td>
<td>E, J</td>
</tr>
<tr>
<td>PRECBASE2 baseline2</td>
<td>For CHANGE MANAGER, specifies the name of baseline 2 that the components use in the pre-Execution comparison of previously created baselines</td>
<td>E, J</td>
</tr>
<tr>
<td>PRECCDL dataSetName</td>
<td>For CHANGE MANAGER, specifies the data set name for the CDL from a pre-Execution step</td>
<td>J</td>
</tr>
<tr>
<td>Keyword</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>PRECDIAG [SYSOUT, dataSetName]</td>
<td>For CHANGE MANAGER, specifies a location for a pre-Execution comparison diagnostic file</td>
<td>E, J</td>
</tr>
<tr>
<td>PRECDL dataSetName</td>
<td>For CHANGE MANAGER, specifies the data set name for the CDL from a pre-Execution comparison</td>
<td>E, J</td>
</tr>
</tbody>
</table>
| PRECOMPARE [B,C] | For CHANGE MANAGER, specifies a pre-Execution comparison  
The following values are valid:  
■ B specifies a baseline-to-baseline comparison.  
■ C specifies a baseline-to-catalog comparison. | E, J |
| PRIPROFILE | For CHANGE MANAGER, specifies whether a profile was used in the JCL for a comparison | J |
| PRODUCT productName | Defines the product name  
The corresponding installation option is PRODUCT. | A, B, C, E, F, I, J, S |
| PROFLOCATION dataSetName | For CHANGE MANAGER, specifies the location of a migrate profile for the JCL for a comparison | J |
| PROFNAME migrateProfile | For CHANGE MANAGER, specifies the name of a migrate profile for the JCL for a comparison | J |
| PROFOWNER owner | For CHANGE MANAGER, specifies the owner of a migrate profile for the JCL for a comparison | J |
| RECEIVE | For CHANGE MANAGER, identifies an Execution worklist as a receive-type worklist | J |
| RECOVERDOPT installationOptionsModule | Specifies the name of the installation options module for the NGT Recover utility  
The BMC_RECOVER_OPTS keyword in the AJXPOFIN input stream replaces this keyword. | F, J |
| RECV01 parameters | Specifies the remote-copy parameters (parameters) for the COPYDDN keyword  
The corresponding installation option is RECVDD01. | J |
| RECV02 parameters | Specifies the remote-copy parameters (parameters) for the COPYDDN keyword  
The corresponding installation option is RECVDD02. | J |
| RECVSTAK | Specifies to stack recovery data sets on tape | J |
| REORGDOPPT installationOptionsModule | Specifies the name of the installation options module for the REORG PLUS utility  
The BMC_REORG_OPTS keyword in the AJXPOFIN input stream replaces this keyword. | F, J |
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
</table>
| RETAINMAXnnnn  | For CHANGE MANAGER, specifies the number of baselines to retain  
See also the RETAINMAX keyword in the ALUIN input stream.                                                                                                                                                                                                                     | J                                |
| SDSNE dataSetName | Defines the default diagnostic output data set name for Execution  
The corresponding installation option is SDSNE.                                                                                                                                                                                                                          | E, F, J                           |
| SENDONLY       | For CHANGE MANAGER, generates JCL for phase 1 of a two-phase migration                                                                                                                                                                                                                                                                     | J                                |
| SORTNUM n      | Specifies the number of sort work data sets                                                                                                                                                                                                                                                                                                 | J                                |
| SSID ssid      | Identifies the DB2 subsystem ID or the DB2 data sharing group attachment name  
The SSID keyword must match the -SSID command in the worklist. This parameter is required.  
If the SSID is specified in the JCL in an EXEC statement in a cataloged procedure (which begins with a PROC statement), its value overrides the value of the SSID keyword.  
The AEXIN and ALUIN input streams also use this keyword. | A, B, BR, E, F, I, J, P, S       |
| SUBMIT         | For the BMCTRIG component of DASD MANAGER PLUS, specifies whether to submit JCL                                                                                                                                                                                                                                                             | N/A                              |
| SYSCSTAK       | Stacks the local image-copy data sets on tape                                                                                                                                                                                                                                                                                                | J                                |
| SZDEVT [3380,3390] | Specifies the device type used in data set sizing  
Device type 3380 is the default and should be used if the device type is mixed or unknown.      
The corresponding installation option is SZDEVT.                                                                                   | F, J                              |
| UNLDSTAK       | Stacks the unload (SYSREC) data sets on tape                                                                                                                                                                                                                                                                                                 | J                                |
| UNLOADDOPT installationOptionsModule | Specifies the name of the UNLOAD PLUS utility installation options module  
The BMC_UNLOAD_OPTS keyword in the AJXPOFIN input stream replaces this keyword.                                                                                                           | F, J                              |
| UTILITYID utilityID | For the BMCTRIG component of DASD MANAGER PLUS, identifies the utility ID that is specified in Execution JCL                                                                                                           | N/A                              |
| VCAT           | For DASD MANAGER PLUS, identifies the VCAT for jobs  
The AEXIN input stream also uses this keyword.                                                                                                                                                                                                                                  | N/A                              |
| VERS7FMT       | Specifies whether Batch Execution JCL Generation uses the POF                                                                                                                                                                                                                                                                                 | J                                |
### ALUIN keywords

Various components of the ALTER and CHANGE MANAGER products use the ALUIN input stream.

Table 16 on page 72 lists the keywords in the ALUIN input stream. For the list of components to which the **Used by** column refers, see “Overview of the keyword tables” on page 54.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKID <em>a,b</em></td>
<td>Specifies the work ID to use Execution fails if this work ID does not match the work ID that the -WKID command in the worklist specifies. The AEXIN and ALUIN input streams also use this keyword.</td>
<td>A, B, E, I</td>
</tr>
<tr>
<td>WORKLISTDDN <em>ddname</em></td>
<td>For the BMCTRIG component of DASD MANAGER PLUS, specifies the alternate DD for the worklist</td>
<td>N/A</td>
</tr>
<tr>
<td>WORKLISTFORMAT</td>
<td>For the BMCTRIG component of DASD MANAGER PLUS, specifies the standard or worklist JCL</td>
<td>N/A</td>
</tr>
<tr>
<td>WPS</td>
<td>Defines the default work primary space allocation The corresponding installation option is WPS.</td>
<td>F, J</td>
</tr>
<tr>
<td>WSS</td>
<td>Defines the default work secondary space allocation The corresponding installation option is WSS.</td>
<td>F, J</td>
</tr>
</tbody>
</table>
### Table 16: ALUIN keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLOWICP</td>
<td>STOPICP=N DUAL=N</td>
<td>Specifies that index-controlled partitioned tables can be created. This keyword overrides the STOPICP=Y or DUAL=Y installation options and overrides the ZPARM PREVENT_NEW_IXCTRL_PART = YES. This keyword must be manually added in the ALUIN input stream. See also the STOPICP keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>ALLOWREBUILD (DB2 Version 11 and later)</td>
<td>None</td>
<td>If Analysis generates message BMC396319E, manually add this keyword in the ALUIN input stream to allow Analysis to generate UNLOAD, DROP, CREATE, and LOAD statements in a worklist for a table space. This keyword is valid only if the limit key values on the current last partition and the added partition or partitions overlap. <strong>WARNING:</strong> Use this keyword if data exists in the overlap range to ensure that no data loss occurs in the current last partition.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>ALLOWREORG (DB2 Version 11 and later)</td>
<td>None</td>
<td>If Analysis generates message BMC396318E, manually add this keyword in the ALUIN input stream to allow Analysis to generate a REORG statement and an ALTER statement to add or rotate partitions for a table space. This keyword must be manually added in the ALUIN input stream. This keyword is valid only if the limit key values on the current last partition and the added partition or partitions overlap. <strong>WARNING:</strong> Do not add this keyword if data exists in the overlap range, as data will be lost from the existing last partition. Keyword only valid for DB2 V11 and later.</td>
<td>A</td>
</tr>
<tr>
<td>ALLSTATSUPD</td>
<td>UPDSTATS=A</td>
<td>Specifies that the tables DASD MANAGER PLUS and the DB2 catalog tables should be updated.</td>
<td>A</td>
</tr>
<tr>
<td>ALTERADDVOL</td>
<td>None</td>
<td>Causes space to be allocated only for the first volume in the IDCAMS DEFINE CLUSTER command. To allow for additional candidate volumes, ALTER or CHANGE MANAGER uses the IDCAMS ALTER ADDVOLUME command. This keyword is useful if you use the Storage Management System (SMS) with &quot;guaranteed space.&quot; You must manually add this keyword to the ALUIN input stream for a worklist.</td>
<td>A</td>
</tr>
<tr>
<td>AMSDELETEI</td>
<td>None</td>
<td>Inserts a DELETE statement for the &quot;I&quot; data set that stores reorganized data structures and data after an online reorg. See also the NOAMSDELETEI keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>AUTHSW</td>
<td>AUTHSW=Y</td>
<td>Controls the method of authorization ID switching that Analysis uses. The products generate -AUTH commands in the worklist. See also the AUTHSWGLID, AUTHSWOFF, and NOAUTHSW keywords in this table. Do not use the AUTHSW keyword if you are using a global authorization ID (GLID) or if AUTHSW=N in the installation options module.</td>
<td>A</td>
</tr>
<tr>
<td>AUTHSWGLID</td>
<td>AUTHSW=G</td>
<td>Disables authorization switching. The products do not generate -AUTH and -SETS commands; they do, however, generate -GLID commands in the worklist. See also the AUTHSW, AUTHSWOFF, and NOAUTHSW keywords in this table.</td>
<td>A</td>
</tr>
<tr>
<td>AUTHSWOFF</td>
<td>AUTHSW=X</td>
<td>Disables authorization switching. Use this keyword only if you are not allowed to execute any -AUTH or -SETS commands in your environment, or if you previously edited your worklists and removed all of the -AUTH, -SETS, and -GLID commands. See also the AUTHSW, AUTHSWGLID, and NOAUTHSW keywords in this table.</td>
<td>A</td>
</tr>
<tr>
<td>BASICUNLOAD</td>
<td>BMCUNLD=N</td>
<td>Generates commands for the UNLOAD PLUS utility.</td>
<td>A</td>
</tr>
<tr>
<td>BINDONIX</td>
<td>BINDONIX=Y</td>
<td>Automatically rebinds all packages dependent upon a table, after a new index has been added to a table or an existing index has been altered.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>BLNAME a.b</td>
<td>None</td>
<td>Passes a baseline name or template to the component if a template from the baseline profile is not being used.</td>
<td>A, B</td>
</tr>
<tr>
<td>BLPROFILE a.b</td>
<td>None</td>
<td>Passes a baseline profile name to the component.</td>
<td>A, B</td>
</tr>
<tr>
<td>BLRECOVER</td>
<td>None</td>
<td>Generates a worklist for restoring a full-recovery baseline. This parameter generates LOAD statements from the baseline unload data sets and will not generate UNLOAD statements.</td>
<td>A</td>
</tr>
<tr>
<td>BLRECOVERPOINT</td>
<td>None</td>
<td>Generates a worklist for establishing a full-recovery baseline. In addition to including the -BASE command in the worklist, this option also generates -UNLD and -UNRC commands to unload and record the baseline unload data sets.</td>
<td>A, B</td>
</tr>
<tr>
<td>BLWORKID a.b</td>
<td>None</td>
<td>Passes the alter-type work ID that established the baseline to Compare. Compare uses the work ID to look up the name of the baseline.</td>
<td>C</td>
</tr>
<tr>
<td>BMCCHECK</td>
<td>BMCCHECK=Y</td>
<td>Uses the CHECK PLUS utility for checking referential constraint violations.</td>
<td>A</td>
</tr>
<tr>
<td>BMCCOPY</td>
<td>BMCCOPY=Y</td>
<td>Uses the NGT Copy utility for creating image copies.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>BMCFASTLOAD</td>
<td>BMCFASTL=Y</td>
<td>Uses the FORMAT BMCLOAD option in the UNLOAD PLUS utility and the FORMAT BMCUNLOAD option in the LOADPLUS utility to unload data from one table and load it into another table that has a similar structure. You cannot use spanned records when BMCFASTLOAD is enabled. A worklist cannot perform an unload with BMCFASTLOAD enabled using UNLOAD PLUS on tables supporting spanned columns. CHANGE MANAGER automatically disables BMCFASTLOAD in these circumstances. To override this keyword, you must manually change the keyword to NOBMCFASTLOAD in the ALUIN input stream. See also the NOBMCFASTLOAD keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>BMCLOAD</td>
<td>BMCLOAD=Y</td>
<td>Uses the LOADPLUS utility to load data.</td>
<td>A</td>
</tr>
<tr>
<td>BMCREBUILD</td>
<td>REBLD=B</td>
<td>Uses the NGT Recover utility to run against an index, if the index is created with DEFER YES</td>
<td>A</td>
</tr>
<tr>
<td>BMCREORG</td>
<td>REORG=B</td>
<td>Uses the REORG PLUS utility to reorganize table spaces and indexes. If changes to the object attributes cause DB2 to place the object in a REORG-pending (AREO*) advisory status or in a REORG-pending (REORP) restrictive status, ALTER and CHANGE MANAGER include the -BMCR reorganization command in a worklist.</td>
<td>A</td>
</tr>
<tr>
<td>BMCSTATSUPD</td>
<td>UPDSTATS=B</td>
<td>Specifies that only the DASD MANAGER PLUS tables are updated.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>BMCUNLOAD</td>
<td>BMCUNLD=Y</td>
<td>Uses the UNLOAD PLUS utility to unload data.</td>
<td>A</td>
</tr>
<tr>
<td>BUILDTEMPCOPY</td>
<td>None</td>
<td>Builds a portable copy of the image copy data set. See also the NOBUILDTEMPCOPY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>CATALOG</td>
<td>AMS=C</td>
<td>Generates the following statements and commands in a worklist:</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ AMS statements, including the CATALOG parameter in the IDCAMS DEFINE CLUSTER statement.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Commands for specifying IBM Storage Management Subsystem (SMS) parameters.</td>
<td></td>
</tr>
<tr>
<td>CDLCHANGERULES a.b</td>
<td>None</td>
<td>Specifies the name of an outbound migrate profile containing change rules that are applied to the Change Definition Language (CDL) after Compare performs a comparison. This keyword replaces the MIGPROFILE keyword in the Compare component of CHANGE MANAGER.</td>
<td>C</td>
</tr>
<tr>
<td>CHANGERULESIN2 migrateProfile</td>
<td>None</td>
<td>Specifies the name of an outbound migrate profile containing change rules that Compare uses to resolve object names and creators. Compare applies the change rules to CMPIN2 objects before the component performs a comparison. The outbound migrate profile must not contain locations. This keyword replaces the CMPPROFILE keyword.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>CISIZE4K</td>
<td>None</td>
<td>For VCAT-defined table spaces (partitioned, simple, segmented, and large object, or LOB), creates VSAM data sets with a control interval size (CISIZE) of 4 KB, regardless of the page size (buffer pool size) of the table space. The CISIZE4K keyword overrides the value that you specify for the Variable CISIZE field on the Tablespace Parts List panel or the Auxiliary Tablespace Attributes List panel. For partitioned table spaces, the product uses a 4 KB CISIZE for each partition of the table space. You cannot specify CISIZE4K for only certain partitions in a table space.</td>
<td>A</td>
</tr>
<tr>
<td>CKEXCEPTIONS <em>nnnn</em></td>
<td>EXCEPT=0</td>
<td>Specifies the maximum number of rows that contain referential integrity or check constraint errors. The default is 0. You must manually add this keyword to the ALUIN input stream.</td>
<td>A</td>
</tr>
<tr>
<td>CLONEDATA</td>
<td>CLONEDATA= Y</td>
<td>Includes worklist commands to unload, load, and copy data contained in clone tables. See also the NOCLONEDATA keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>CMPILLOT</td>
<td>None</td>
<td>Specifies that the Analysis job contains CM/PILOT Data Manipulation Language (DML)</td>
<td>A, P</td>
</tr>
<tr>
<td>CMPIN1 *a.b</td>
<td>None</td>
<td>Passes the name of the primary set of input (a baseline or a work ID) in a comparison to Compare.</td>
<td>C</td>
</tr>
<tr>
<td>CMPIN2 *a.b</td>
<td>None</td>
<td>Passes the name of the secondary set of input (baseline) in a comparison to Compare.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>CMPTYPE1 <em>inputType</em></td>
<td>None</td>
<td>Passes the type of the primary set of input to Compare. Valid types of input are BASELINE, BLWORKID, DDL, LOCAL (DB2 catalog), REMOTE (DB2 catalog), WORKID, and WORKLIST.</td>
<td>C</td>
</tr>
<tr>
<td>CMPTYPE2 <em>inputType</em></td>
<td>None</td>
<td>Passes the type of the secondary set of input to Compare. Valid types of input are BASELINE, BLWORKID, DDL, LOCAL (DB2 catalog), REMOTE (DB2 catalog), and WORKLIST.</td>
<td>C</td>
</tr>
<tr>
<td>COMMIT <em>n</em></td>
<td>None</td>
<td>Generates an SQL COMMIT statement after <em>n</em> SQL INSERT statements when CHANGE MANAGER creates a baseline. The default is 500. Valid values are 1 through 2,147,483,647.</td>
<td>B</td>
</tr>
<tr>
<td>COPYDDN (<em>parameters</em>)</td>
<td>COPYDD01=C, COPYDD02=C, RECVDD01=C, RECVDD02=C</td>
<td>Defines image copies for the IBM COPY and LOAD utilities and the NGT Copy, LOADPLUS, REORG PLUS, and NGT Recover utilities. Local-copy parameters (<em>parameters</em>) for the COPYDDN keyword are COPY01 and COPY02. Separate the parameters with commas or blanks. Remote-copy parameters (<em>parameters</em>) for the COPYDDN keyword are RECV01 and RECV02. Separate the parameters with commas or blanks.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------</td>
<td>-------------</td>
<td>----------------------------------</td>
</tr>
</tbody>
</table>
| COPYEXPORT | None | The Copy Migration feature uses the NGT Copy EXPORT command and the NGT Recover IMPORT command to migrate an image copy or set of image copies to another DB2 subsystem. The Copy Migration feature is included in the following solutions:  
- Recovery Management  
- BMC Recovery for DB2  
- BMC Object Administration for DB2  
- BMC Next Generation Technology Database Administration for DB2  
- Database Administration | A |
<p>| CVALOFF | None | Ignores Change Definition Language (CDL) statements that reference nonexistent table and view columns. Any CDL statements that specify dropping or altering nonexistent table and view columns are discarded when the CDL is imported. If the CDL specifies adding a new column before or after one that does not exist, the new column is placed at the end of the column list. | I |
| DB2STATSUPD | UPDSTATS=C | Specifies that only the DB2 catalog tables should be updated | A |</p>
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBRMLIB</td>
<td>DBRMLIB</td>
<td>Includes the LIBRARY parameter on the BIND statement for plans and packages. A disadvantage to adding the LIBRARY parameter to the BIND PLAN statement is that the order of the libraries on the BIND might be incorrect. If some DBRMs are present in multiple libraries, ALTER and CHANGE MANAGER cannot guarantee that the concatenation will result in every DBRM coming from the correct library. In addition, the LIBRARY parameter overrides any DBRM library concatenation in the JCL.</td>
<td>A</td>
</tr>
<tr>
<td>DEBUG</td>
<td>None</td>
<td>Causes Analysis and CM/PILOT to print complete diagnostic information in the diagnostic output file. This parameter might be necessary if you suspect that Analysis or CM/PILOT is operating incorrectly and you need to send supporting documentation to BMC for research. <strong>Note:</strong> Analysis also uses the FULLDEBUG keyword.</td>
<td>A, P</td>
</tr>
<tr>
<td>DEBUG(INIT)</td>
<td>None</td>
<td>Causes a CHANGE MANAGER component to print complete diagnostic information on the diagnostic output file. This parameter might be necessary if you suspect that one of the components is operating incorrectly and you need to send supporting documentation to BMC for research.</td>
<td>B, BR, C, I</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>DEFAULTOFF</td>
<td>None</td>
<td>Retains the name of the owner of an object or the database name for a table space or table when CHANGE MANAGER cannot determine the correct name. If this keyword is not specified for new (created) objects, the owner or database name is replaced with a set of pound (#) signs. This keyword is necessary only in a comparison of one DB2 catalog to another DB2 catalog in which an outbound migrate profile (CHANGERULESIN2) is not specified.</td>
<td>C</td>
</tr>
<tr>
<td>DEFERUNIQUEIX</td>
<td>DEFERUIX=Y</td>
<td>Specifies to create new unique indexes with the DEFER YES parameter.</td>
<td>A</td>
</tr>
<tr>
<td>DELETEAGE $n$</td>
<td>None</td>
<td>Specifies deleting baselines that are older than the current date (minus $n$ days) and created with the specified baseline profile. See also the DELETEAGE keyword in the AJXIN input stream.</td>
<td>B</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>DISCARDS $n$</td>
<td>DISCARDS=$nnnn$</td>
<td>Specifies the number of discard records to allow. The valid range for $n$ is 0 through 9999. DISCARDS 0 means that no maximum number of discards exists. If you use DISCARDS 1, the product generates one discard DD, a //SYSDS001 for the entire run, and DISCARDS 1 as a LOAD parameter. JCL generated minimally sizes the data sets for SYSDS001 and SYSER001 DDs. This action causes the load utility to terminate with return code 8 if any records must be discarded. Typically, you use DISCARDS 1 when data is migrated, valid data remains unchanged, and the goal is to reduce the number of DDs and data sets needed. If the DISCARDS keyword is set to anything other than 1, the product generates a different discard DD (//SYSD$nnnn$) for each LOAD, and generates DISCARDS $n$ as a LOAD parameter for each LOAD command (where $n$ is the maximum number of discards). This action causes the load to terminate if the discard maximum is reached. If fewer records are discarded, the discard file contains the records and execution proceeds to the next step in the worklist. You must manually add this keyword to the ALUIN input stream to change the number of discard records set in the DISCARDS installation option.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
</tbody>
</table>
| DYNCOPY   | DYNCOPY=Y                          | Dynamically allocates the copy (SYSCOPY) data sets that the NGT Copy, NGT Recover, and IBM COPY utilities use to create image copies.  
If you specify DYNCOPY, you cannot specify to use noncopy utilities to build image copies with the UTILCOPY keyword. These noncopy utilities include the LOADPLUS and REORG PLUS utilities, and the IBM LOAD and REORG utilities.  
NGT Reorg always uses dynamic allocation and disregards this keyword.  
See also the NODYNCOPY keyword in this table.                                                                                                                                                                                                                      | A, F                             |
| DYNREORG  | DYNREORG=Y                         | Dynamically allocates the archive (ARCHDDN), copy (COPYDDN, RECOVERYDDN), discard (DISCARDDDN), punch (PUNCHDDN), sysrec (UNLDDN), and sortout or sysut (WORKDDN) data sets that the REORG PLUS and IBM REORG utilities use.  
**Note:** NGT Reorg always dynamically allocates data sets, disregards this keyword and disregards the DYNREORG installation option.  
See also the NODYNREORG keyword in this table.                                                                                                                                                                                                               | A, F                             |
| DYNUNLD   | DYNUNLD=Y                          | Dynamically allocates the unload (SYSREC) and discard (SYSDISC) data sets that the UNLOAD PLUS, LOADPLUS, IBM UNLOAD, and IBM LOAD utilities use.  
DYNUNLD=Y is required when you generate a worklist with spanned record column support.  
See also the NODYNUNLD keyword in this table.                                                                                                                                                                                                             | A, F                             |
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMPTYCARD n</td>
<td>None</td>
<td>Regards tables with a specific cardinality to be considered empty. The valid range for n is 0 through 2,147,483,647. In certain applications, particularly ERP applications, manually updating statistics to get a nonzero cardinality for tables that might be empty can improve performance. When optimizing ERP applications, NOUNLOADEMPTY views all tables with a cardinality of zero and the cardinality specified by EMPTYCARD as empty. EMPTYCARD n must be manually added into the ALUIN input stream, and the NOUNLOADEMPTY parameter must also be included in the input stream.</td>
<td>A</td>
</tr>
<tr>
<td>ENV</td>
<td>None</td>
<td>Prints ALTER or CHANGE MANAGER environment information (including a list of indexes that the components use) in its diagnostic output. The ENV keyword also includes a list of the keywords and values in the installation options module, and the values for the DSNZPARM parameters. The AEXIN input stream also uses this keyword.</td>
<td>A, B, BR, C, E, F, I, J</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------</td>
<td>-------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>EURO</td>
<td>EURO</td>
<td>Expects numbers in the European format (comma used for the decimal point) and create output in European decimal format. This parameter is particularly important when ALTER or CHANGE MANAGER parses index LIMITKEY values that are separated by commas. If the EURO keyword is present, ALTER or CHANGE MANAGER requires delimiting commas to be followed by blanks. The Import, Specification, Baseline, and Compare components use the value for EURO from the installation options module but do not support use of the EURO keyword in the ALUIN input data stream. The AJXIN input stream also uses this keyword.</td>
<td>A, B, BR, C, I, J, S</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
</tbody>
</table>
| EXPMAXTS \( nnnn \) | None                               | For the Copy Migration feature, specifies the number of table spaces to include in a single migration file that the NGT Copy EXPORT command creates. The valid range of values is 0 through 9999. The default value is 100. A value of 0 indicates that all table spaces in the scope will be included in a single file. The Copy Migration feature is included in the following solutions:  
  - Recovery Management  
  - BMC Recovery \textit{for DB2}  
  - BMC Object Administration \textit{for DB2}  
  - BMC Next Generation Technology Database Administration \textit{for DB2}  
  - Database Administration | A |
<p>| FORCECHECK      | None                               | Uses the ENFORCE NO option when LOADPLUS loads data and uses the IBM CHECK DATA utility to enforce referential integrity and check constraints.                                                              | A                               |</p>
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
</table>
| FORCELOADREPLACE        | None                              | Instructs Analysis to generate one of the following forms of the LOAD DATA syntax:  
  - LOAD DATA RESUME YES INTO TABLE  
    Analysis uses this syntax in the BMC LOADPLUS utility when you are migrating only data from a single table at the table level of a segmented table space.  
    Analysis uses this syntax in the IBM LOAD utility when you are migrating only data from:  
      - A single table at the table level of a segmented table space.  
      - The second and succeeding tables of multiple tables at the table space level of a segmented table space.  
    Analysis does not assume that the structures on the sending and receiving subsystems are identical. The load utility deletes the existing data in the table before loading. | A                                |
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
</table>
| FORCELOADREPLACE (continued) | None | ■ LOAD DATA RESUME YES INTO TABLE REPLACE  
Analysis uses this syntax in the LOADPLUS utility when you are migrating only data from one or more tables at the table space level of a segmented table space. Analysis assumes that the structures on the sending and receiving subsystems are identical, and generates the LOAD statements based on the structure of the sending subsystem. The load utility replaces the data in the table. Analysis does not use this syntax in the IBM LOAD utility. | |
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
</table>
| FORCELOADREPLACE (continued)  | None                              | - LOAD DATA REPLACE INTO TABLE  
Analysis uses this syntax in the LOADPLUS utility when you are migrating only data from an entire table space at the table space level of a partitioned table space.  
Analysis uses this syntax in the IBM LOAD utility when you are migrating only data from:  
— A single table at the table space level of a segmented table space  
— The first table of multiple tables at the table space level of a segmented table space  
— An entire table space at the table space level of a partitioned table space  
Analysis assumes that the structures on the sending and receiving subsystems are identical, and generates the LOAD statements based on the structure of the sending subsystem. The load utility replaces the data in the table space.  
**WARNING:** A high possibility for data loss exists if the structures are not identical.  
You must manually add this keyword to the ALUIN input stream. | A |

---

**90 ALTER and CHANGE MANAGER for DB2 Reference Manual**
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORCERESMAX</td>
<td>None</td>
<td>Uses the NGT Recover product to migrate data from image copy data sets, even if the source data is not in reordered row format (RRF). You must manually add this keyword to the ALUIN input stream for a worklist.</td>
<td>A</td>
</tr>
<tr>
<td>FULLDEBUG</td>
<td>None</td>
<td>Prints complete diagnostic information on the diagnostic output file. This parameter might be necessary if you suspect that Analysis is operating incorrectly and you need to send supporting documentation to BMC for research.</td>
<td>A</td>
</tr>
<tr>
<td>GENOBID</td>
<td>GENOBID=Y</td>
<td>Generates the OBID parameter in a CREATE TABLE statement. See also the NOGENOBID keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>GLAUTHID globalAuthorizationID</td>
<td>GLID</td>
<td>Defines a global authorization ID (GLID). The components use this authorization ID instead of the authorization ID of the person who submits the job. The worklist begins with a -GLID command that switches authorization to the GLID. After all Analysis-generated -AUTH and -SETS commands, the product resets the current authorization ID to the GLID. See also the GLID keyword in the AJXIN input stream.</td>
<td>A, B</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>HISTORY ALL</td>
<td>STATHIST=Y</td>
<td>Uses the IBM RUNSTATS utility to update the DB2 catalog history tables with the current statistics that are being collected. See also the NOHISTORY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>HSMVOL volumeID</td>
<td>HSMVOL=volumeID</td>
<td>Specifies the volume ID that indicates an archived data set if you are using a storage management system, such as the IBM Hierarchical Storage Manager (DFSMShsm). If this volume ID is encountered, ALTER or CHANGE MANAGER uses a template of default values for data set allocation. See also the NOHSMVOL keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>IBMCHECK</td>
<td>BMCCHECK=N</td>
<td>Uses the IBM CHECK DATA utility for checking referential constraint violations.</td>
<td>A</td>
</tr>
<tr>
<td>IBMCOPY</td>
<td>BMCCOPY=N</td>
<td>Uses the IBM COPY utility for creating image copies.</td>
<td>A</td>
</tr>
<tr>
<td>IBMLOAD</td>
<td>BMCLOAD=N</td>
<td>Uses the IBM LOAD utility to load data.</td>
<td>A</td>
</tr>
<tr>
<td>IBMREBUILD</td>
<td>REBLD=I</td>
<td>Uses the IBM REBUILD utility to run against an index if the index is created with DEFER YES.</td>
<td>A</td>
</tr>
<tr>
<td>IBMREORG</td>
<td>REORG=I</td>
<td>Uses the IBM REORG utility to reorganize table spaces and indexes. If changes to the object attributes cause DB2 to place the object in a REORG-pending (AREO*) advisory status or in a REORG-pending (REORP) restrictive status, ALTER and CHANGE MANAGER include the -REOR reorganization command in a worklist.</td>
<td>A</td>
</tr>
<tr>
<td>IBMUNLOAD</td>
<td>BMCUNLD=I</td>
<td>Uses the IBM UNLOAD utility to unload data.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>IDAAIGNORE</td>
<td>None</td>
<td>When you attempt to modify an IDAA table, ALTER and CHANGE MANAGER issue an error message and do not generate the worklist. IDAAIGNORE tells the product to issue a warning message and generate the worklist. If this keyword is included, standard IDAA tables are modified, but IDAA Type D tables and the associated table spaces are not modified. The database is modified for all IDAA tables.</td>
<td>A</td>
</tr>
</tbody>
</table>
| IGNOREIMPLDEPS  | None                              | When you make changes that cause the following objects to be dropped and created, or migrated, ignores the changes to those objects:  
- Table spaces (whether the changes were made in the current work ID or in a previous work ID)  
- Auxiliary objects  
- Enforcing indexes defined as ROWID GENERATED BY DEFAULT  
These objects are dependents of implicitly created table spaces. See also the NOIGNOREIMPLDEPS keyword in this table. | A                                |
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGNORELKERRORE</td>
<td>LKVALID=I</td>
<td>Ignores any errors that the product generates while assigning a new sequence for the ALTER TABLE ALTER PARTITION statements to alter limit keys. You must manually insert this keyword in the ALUIN input stream. Analysis generates the statements in the original order and issues a warning message.</td>
<td>A</td>
</tr>
<tr>
<td>IMPORT</td>
<td>None</td>
<td>Imports CM/PILOT Data Manipulation Language (DML) statements into a work ID.</td>
<td>A, P</td>
</tr>
<tr>
<td>INCLUDE (parameter (parameter,...))</td>
<td>None</td>
<td>Specifies either the types of commands to include in a worklist or the information to include in a Change Definition Language (CDL) file. The following parameters are valid for Analysis: ■ AMS ■ BIND ■ BINDAUTH ■ DATA ■ SQL ■ REBIND The following parameters are valid for Compare: ■ CDLDEP ■ ALTER ■ CREATE ■ DROP</td>
<td>A, BR, C</td>
</tr>
<tr>
<td>KEYCARD</td>
<td>KEYCARD=Y</td>
<td>Takes cardinality statistics for the key columns of an index when the IBM RUNSTATS utility takes index statistics. See also the NOKEYCARD keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>LOBGT32M</td>
<td>None</td>
<td>For the Database Administration, or BMC Object Administration for DB2 solutions, generates a worklist for large object (LOB) columns that are defined greater than 32 megabytes (MB) in length. You must manually insert this keyword in the ALUIN input stream. In addition, you must select the Use Dynamic SYSRE datasets option on the Analysis Utility Dataset Options panel.</td>
<td>A</td>
</tr>
<tr>
<td>LOCATION1</td>
<td>None</td>
<td>Specifies the location of a DB2 catalog for the primary set of input in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>LOCATION2</td>
<td>None</td>
<td>Specifies the location of a DB2 catalog for the secondary set of input in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>LOG</td>
<td>LOG=Y</td>
<td>Logs the records that the IBM LOAD utility loads.</td>
<td>A</td>
</tr>
<tr>
<td>MAPPINGDATABASE dataBase</td>
<td>None</td>
<td>Specifies the database that will contain the mapping table.</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> You must specify an existing database.</td>
<td></td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
</tbody>
</table>
| MAXSYSREC nnnn      | None                               | Specifies the maximum number of dynamically allocated unload (SYSREC) data sets to use for each unload (-BMCD) and load (-BMCL) worklist command when multitasking the unloading and loading of data for partitioned table spaces. If the number of partitions in the table space exceeds the value of MAXSYSREC, Analysis divides the partitions to be unloaded among several unload and load worklist commands. For example, if the value of MAXSYSREC is 256 and the number of partitions in the table space is 540, Analysis generates three unload and load commands:  
- The first command unloads or loads the first 256 partitions.  
- The second command unloads or loads the next 256 partitions.  
- The third command unloads or loads the last 28 partitions.  
The default value is 256. If you want to specify a value other than 256, you must manually change the value for the keyword in the ALUIN input stream. | A                                |
<p>| MAXSYSUT nnnn       | MAXSYSUT                           | Specifies the maximum number of SYSUT temporary work data sets that the LOADPLUS and REORG PLUS utilities use to build nonclustering indexes. Valid values are 1 through 9999.                                                                                                                                                                                                                                                                         | A                                |</p>
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
</table>
| MAXVER              | None                              | Performs an action if an online schema change cannot be made to an object because the maximum number of active versions exists for the object. Analysis will perform one of the following actions:  
  - Include statements in the worklist to drop and create an object.  
  - For indexes that are defined as COPY NO, generate a -RBLD statement in the worklist to rebuild the index.  
  If the MAXVER keyword is not included in the ALUIN input stream and Analysis encounters an object that has reached the maximum number of active versions, Analysis terminates with an error. You must manually add this keyword to the ALUIN input stream. | A                                |
<p>| MIGLOCATIONS        | location ddname [...])            | Specifies which locations (location) to use in an outbound migrate profile and the associated DDName for the output worklist or Change Definition Language (CDL) file for each location. The location and DDName must be separated by a space, and the location and DDName pairs must be separated by commas. If MIGLOCATIONS is not specified, all locations in the outbound migrate profile are used in Analysis or the comparison. | A, C                             |</p>
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIGPROFILE a.b</td>
<td>None</td>
<td>Passes the name of a migrate profile. The Compare component of CHANGE MANAGER uses the CDLCHANGERULES keyword instead of the MIGPROFILE keyword.</td>
<td>A, I</td>
</tr>
<tr>
<td>MIGSCOPE</td>
<td>None</td>
<td>Migrates from a profile scope rather than from the work ID’s CD entries.</td>
<td>A</td>
</tr>
<tr>
<td>MULTITBDS</td>
<td>None</td>
<td>Generates a single -BMCD worklist command for a table space that contains multiple tables when you use the UNLOAD PLUS utility to unload data. Analysis unloads each table in the table space into in a separate unload (SYSREC) data set. You must manually add this keyword to the ALUIN input stream. In addition, you must select the Use Dynamic SYSREC datasets option on the Analysis Utility Dataset Options panel.</td>
<td>A</td>
</tr>
<tr>
<td>NEWWORKID a.b</td>
<td>None</td>
<td>Creates a new work ID. To use this keyword, Analysis also requires the MIGPROFILE a.b and MIGSCOPE keywords. The AEXIN input stream also uses this keyword.</td>
<td>A, I, P</td>
</tr>
<tr>
<td>NOALIAS</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the alias object.</td>
<td>C</td>
</tr>
<tr>
<td>NOAMSDELETEI</td>
<td>None</td>
<td>Does not delete an &quot;I&quot; data set that contains data structures and data that have been reorganized in an online reorg. See also the AMSDELETEI keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>NOAUTHSW</td>
<td>AUTHSW=N</td>
<td>Uses only the SET CURRENT SQLID statement for authorizations. The products generate -SETS commands in the worklist. See also the AUTHSW and AUTHSWOFF keywords in this table. Do not use the NOAUTHSW keyword if you are using a global authorization ID (GLID).</td>
<td>A</td>
</tr>
<tr>
<td>NOAUXILIARY</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the auxiliary object.</td>
<td>C</td>
</tr>
<tr>
<td>NOBASELINE</td>
<td>None</td>
<td>Does not create a baseline when using the specified baseline profile.</td>
<td>B</td>
</tr>
<tr>
<td>NOBINDONIX</td>
<td>BINDONIX=N</td>
<td>Does not rebind packages that are dependent upon a table after an index has been added to the table or an existing index has been altered</td>
<td>A</td>
</tr>
<tr>
<td>NOBMCFASTLOAD</td>
<td>BMCFASTL=N</td>
<td>Does not use the FORMAT BMLOAD option in the UNLOAD PLUS utility and the FORMAT BMCUNLOAD option in the BMC LOADPLUS utility to unload data from one table and load it into another table that has a similar structure. To override this keyword, you must manually change the keyword to BMCFASTLOAD in the ALUIN input stream. See also the BMCFASTLOAD keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOBUFFERPOOL</td>
<td>None</td>
<td>Ignores the BUFFERPOOL attribute for databases, table spaces, and indexes.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>NOBUILDTEMPCOPY</td>
<td>None</td>
<td>Does not build a portable copy of the image copy data set. See also the BUILDTEMPCOPY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOCHECKCNST</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the check constraint object.</td>
<td>C</td>
</tr>
<tr>
<td>NOCLONEDATA</td>
<td>CLONDATA=N</td>
<td>Omits worklist commands to unload, load, and copy data contained in clone tables. See also the CLONEDATA keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOCOMPRESS</td>
<td>None</td>
<td>Ignores the table space COMPRESS attribute from a comparison when you include partition attributes in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>NOCOPY</td>
<td>BMCCOPY=X</td>
<td>Does not create image copies.</td>
<td>A</td>
</tr>
<tr>
<td>NOCOPYFORCE</td>
<td>BMCCOPY=F</td>
<td>Does not create image copies. This parameter causes the product to issue a START DATABASE ACCESS (FORCE) statement to reset the copy pending.</td>
<td>A</td>
</tr>
<tr>
<td>NODATABASE</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the database object or partition attribute. This keyword suppresses comparing object types and object attributes.</td>
<td>C</td>
</tr>
<tr>
<td>NODBRMLIB</td>
<td>None</td>
<td>Omits the LIBRARY parameter on the BIND statement for plans and packages.</td>
<td>A</td>
</tr>
<tr>
<td>NODEFDEFINE</td>
<td>None</td>
<td>Ignores the DDL DEFINE parameter for table spaces and indexes.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>NODROPORPHANEDTS</td>
<td>None</td>
<td>Does not drop any auxiliary table space that does not contain an auxiliary table. You must manually insert this keyword in the ALUIN input stream.</td>
<td>A</td>
</tr>
<tr>
<td>NODROPRESTRICT</td>
<td>None</td>
<td>Ignores the DROPRESTRICT attribute for tables.</td>
<td>C</td>
</tr>
<tr>
<td>NODYNCOPY</td>
<td>DYNCOPY=N</td>
<td>Does not dynamically allocate the copy (SYSCOPY) data sets that the NGT Copy, NGT Recover, and IBM COPY utilities use to create image copies.</td>
<td>A, F</td>
</tr>
<tr>
<td>NODYNREORG</td>
<td>DYNREORG=N</td>
<td>Does not dynamically allocate the archive (ARCHDDN), copy (COPYDDN, RECOVERYDDN), discard (DISCARDDN), punch (PUNCHDDN), sysrec (UNLDDN), and sortout or sysut (WORKDDN) data sets that the REORG PLUS and IBM REORG utilities use. Note: NGT Reorg always dynamically allocates data sets, and disregards this keyword and the DYNREORG installation option. The product allocates the data sets in the Execution JCL.</td>
<td>A, F</td>
</tr>
<tr>
<td>NODYNUNLD</td>
<td>DYNUNLD=N</td>
<td>Does not dynamically allocate the unload (SYSREC) and discard (SYSDISC) data sets that the UNLOAD PLUS, LOADPLUS, IBM UNLOAD, and IBM LOAD utilities use.</td>
<td>A, F</td>
</tr>
<tr>
<td>NOECHO</td>
<td>None</td>
<td>Does not include keywords in the ALUIN input stream as comments in a worklist. You must manually insert this keyword in the ALUIN input stream.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>NOFOREIGNKEY</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the foreign key object.</td>
<td>C</td>
</tr>
<tr>
<td>NOFREEPAGE</td>
<td>None</td>
<td>Ignores the table space and index FREEPAGE attribute from a comparison when you include partition attributes in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>NOGENOBID</td>
<td>GENOBID=N</td>
<td>Does not generate the OBID parameter in a CREATE TABLE statement. See also the GENOBID keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOHISTORY</td>
<td>STATHIST=N</td>
<td>Does not update the DB2 catalog history tables with the current statistics that are being collected. See also the HISTORYALL keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOHSMVOL</td>
<td>HSMVOL=N</td>
<td>Uses a blank value for the HSM volume ID. See also the HSMVOL keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOIDENTITY</td>
<td>None</td>
<td>Ignores the IDENTITY attribute for tables.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
</tbody>
</table>
| NOIGNOREIMPLDEPS         | None                              | Issues an error when you make changes that cause the following objects to be dropped and created, or migrated:  
  - Table spaces (whether the changes were made in the current work ID or in a previous work ID).  
  - Auxiliary objects  
  - Enforcing indexes defined as ROWID GENERATED BY DEFAULT.  
  These objects are dependents of implicitly created table spaces.  
  You must manually add this keyword to the ALUIN input stream.  
  See also the IGNOREIMPLDEPS keyword in this table. | A                                |
| NOINDEX                  | None                              | Sets the object change flag for a comparison to N (no) for the index object.                                                                                                                                 | C                                |
| NOORDERLK                | LKVALID=N                         | Does not analyze or reorder the sequence of ALTER TABLE ALTER PARTITION statements that modify limit keys.  
  You must manually add this keyword to the ALUIN input stream. | A                                |
| NOKEYCARD                | KEYCARD=N                         | Does not take cardinality statistics for the key columns of an index when the IBM RUNSTATS utility takes index statistics.  
  See also the KEYCARD keyword in this table. | A                                |
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOLOBDATA</td>
<td>None</td>
<td>Does not unload and load data contained in large object (LOB) columns. This keyword is not valid when you are creating a full-recovery baseline. You must manually insert this keyword in the ALUIN input stream.</td>
<td>A</td>
</tr>
<tr>
<td>NOLOCKSIZE</td>
<td>None</td>
<td>Ignores the LOCKSIZE and LOCKMAX attributes for table spaces.</td>
<td>C</td>
</tr>
<tr>
<td>NOMTASK</td>
<td>None</td>
<td>Disables multitasking when data is unloaded to permit tape stacking for Execution. You must manually insert this keyword in the ALUIN input stream.</td>
<td>A</td>
</tr>
<tr>
<td>NOORDERBY</td>
<td>None</td>
<td>Bypasses the ORDER BY clause on all unloads. As a result, the Analysis performs the subsequent load from an unsorted file. See also the ORDERBY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOPARALLEL</td>
<td>PARALLEL=N</td>
<td>For the Database Administration and BMC Object Administration for DB2 solutions, does not create commands that Execution can use to run a worklist in parallel. See also the PARALLEL keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOPARTCOPY</td>
<td>PARTCPY=N</td>
<td>Does not create a partition-level image copy of a partitioned table space or index. See also the PARTCOPY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOPARTITION</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the partition attribute.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>NOPATH</td>
<td>None</td>
<td>Ignores the PATH attribute for check constraints, views, and triggers.</td>
<td>C</td>
</tr>
<tr>
<td>NOPCTFREE</td>
<td>None</td>
<td>Ignores the table space and index PCTFREE attribute from a comparison when you include partition attributes in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>NOPLANMGMT</td>
<td>PLANMGMT=N</td>
<td>Does not use the provided default or provide a new default when a REBIND statement runs.</td>
<td>A</td>
</tr>
<tr>
<td>NOPRIQTY</td>
<td>None</td>
<td>Ignores the table space and index PRIQTY attribute from a comparison when you include partition attributes in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>NOREBUILD</td>
<td>REBLD=N</td>
<td>Does not use the IBM REBUILD or the NGT Recover utility. The product generates a CREATE INDEX statement without a DEFER YES clause.</td>
<td>A</td>
</tr>
<tr>
<td>NOREFRESH</td>
<td>REFRESH=N</td>
<td>Does not use the REFRESH TABLE statement to refresh the data in a materialized query table (MQT). You can manually add this keyword to the ALUIN input stream. You can also the REFRESH keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>NOREGENIDENTITY</td>
<td>None</td>
<td>Does not allow a column that is defined with GENERATED ALWAYS AS IDENTITY to be rebuilt by Analysis if the table is dropped and created. To enable DB2 to generate a new unique value for the identity column that is defined as GENERATED ALWAYS, manually change the NOREGENIDENTITY keyword to REGENIDENTITY. See also the REGENIDENTITY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOREMARKS</td>
<td>None</td>
<td>Ignores the objects that have changes only to comments and labels.</td>
<td>C</td>
</tr>
</tbody>
</table>
| NOREORGONLINE           | REORGONL=N                         | Does not use an online reorg (SHRLEVEL CHANGE) to reorganize a table space. You can use the SHRLEVEL REFERENCE or SHRLEVEL NONE option to reorganize the table space. Either select one of these options on the Analysis Reorg Option panel, or add the relevant keyword to the ALUIN input stream:  
- REORGREF (for SHRLEVEL REFERENCE)  
- REORGNONE (for SHRLEVEL NONE)  

**Note:** NGT Reorg always performs an online reorganization (SHRLEVEL CHANGE), and disregards this keyword. See also the REORGONLINE keyword in this table. | A                                |
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOREPLACE</td>
<td>None</td>
<td>Specifies that existing work IDs will not be replaced with the REPLICA WORKID statement. See also the REPLACE keyword in this table.</td>
<td>P</td>
</tr>
<tr>
<td>NOROLEOWN</td>
<td>ROLEOWN=N</td>
<td>Specifies that a ROLE is not defined as the owner, creator, or grantor of an object. See also the ROLEOWN keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOSECQTY</td>
<td>None</td>
<td>Ignores the table space and index SECQTY attribute from a comparison when you include partition attributes in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>NOSEGSIZE</td>
<td>None</td>
<td>Ignores the SEGSIZE attribute for segmented table spaces.</td>
<td>C</td>
</tr>
<tr>
<td>NOSPANNEDRECS</td>
<td>SPANNED=N</td>
<td>Uses file reference data sets instead of spanned records to unload and load XML and LOB data</td>
<td>A</td>
</tr>
<tr>
<td>NOSTARTVALUE</td>
<td>None</td>
<td>Ignores the START attribute for an identity column in a table.</td>
<td>C</td>
</tr>
<tr>
<td>NOSTOGROUP</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the storage group object or attribute. This keyword suppresses comparing object types and object attributes.</td>
<td>C</td>
</tr>
<tr>
<td>NOSTOPCOMMIT</td>
<td>STOPCOMMIT=N</td>
<td>Prevents an AT(COMMIT) statement from being generated for every STOP command in the worklist. See also the STOPCOMMIT keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>NOSTOPLIST</td>
<td>STOPLIST=creatorName</td>
<td>Analyzes all of the tables in a migrate-type work ID, including those in a list that are identified by a creator name for the work ID as tables that should not be analyzed. See also the STOPLIST keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOSYNONYM</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the synonym object.</td>
<td>C</td>
</tr>
<tr>
<td>NOTABLE</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the table object. Use this keyword to set the flags for global temporary tables.</td>
<td>C</td>
</tr>
<tr>
<td>NOTABLEACCESS</td>
<td>TABLEACC=N</td>
<td>Includes START DATABASE commands in an alter-type worklist, preventing outside access to tables during worklist execution. Whether or not NOTABLEACCESS is specified, the SHRLEVEL NONE option is used as the default in a REORG TABLESPACE statement in a -BMCR or -REOR worklist command.</td>
<td>A</td>
</tr>
<tr>
<td>NOTABLEALL</td>
<td>TABLEALL=N</td>
<td>Does not include the TABLE(ALL) parameter on stand-alone statistics jobs.</td>
<td>A</td>
</tr>
<tr>
<td>NOTABLESPACE</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the table space object or partition attribute. This keyword suppresses comparing object types and object attributes.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>NOTCPROTATE</td>
<td>None</td>
<td>Does not create an ALTER TABLE ROTATE PARTITION statement in a worklist, even if you specified a value for the Rotate Parts field on the Tablespace Parts List panel for the work ID. You must manually add this keyword to the ALUIN input stream. See also the TCPROTATE keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOTTRACKMOD</td>
<td>None</td>
<td>Ignores the table space TRACKMOD attribute from a comparison when you include partition attributes in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>NOTRIGGER</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the trigger object.</td>
<td>C</td>
</tr>
<tr>
<td>NOUNIQUECNST</td>
<td>None</td>
<td>Shows a primary key as part of the table definition in the generated Change Definition Language (CDL). If you omit the keyword, Compare shows a primary key as a primary unique constraint.</td>
<td>C</td>
</tr>
<tr>
<td>NOUNLDCOLL</td>
<td>UNLDCOLL=N</td>
<td>Provides explicit column lists on all of the UNLOAD PLUS utility unloads only when needed. To provide explicit column lists on the IBM UNLOAD utility unloads, you must manually insert the keyword in the ALUIN input stream.</td>
<td>A, J</td>
</tr>
<tr>
<td>NOUNLOADEMPT</td>
<td>UNLDEMT=N</td>
<td>Does not unload tables that the IBM RUNSTATS utility indicates are empty. See also the UNLOADEMPTY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>NOUSENGTR</td>
<td>USENGTR=N</td>
<td>Uses the REORG PLUS utility to perform data reorganization. See also the USENGTR keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOUTILCOPY</td>
<td>UTILCOPY=N</td>
<td>Specifies that an image copy is taken by the IBM COPY utility or the NGT Copy utility, and not by another utility. See also the UTILCOPY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOVALIDATE</td>
<td>None</td>
<td>Does not check for changes in objects since specification was performed.</td>
<td>A</td>
</tr>
<tr>
<td>NOVIEW</td>
<td>None</td>
<td>Sets the object change flag for a comparison to N (no) for the view object.</td>
<td>C</td>
</tr>
<tr>
<td>NOVIEWCHECK</td>
<td>None</td>
<td>Ignores the CHECK attribute for views.</td>
<td>C</td>
</tr>
<tr>
<td>NOVVALPROP</td>
<td>VVALPROP=N</td>
<td>Specifies that extended view text propagation is not supported by ALTER or CHANGE MANAGER. See also the VVALPROP keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>NOWLORDER</td>
<td>None</td>
<td>Generates the objects in a worklist in an unsorted, random order. See also the WLORDER NONE keyword in this table. You can use this keyword to override the ACM_WLORDER POF keyword.</td>
<td>A</td>
</tr>
<tr>
<td>NOXMLDATA</td>
<td>None</td>
<td>Does not unload and load data contained in XML columns. You must manually insert this keyword in the ALUIN input stream. This keyword is not valid when you are creating a full-recovery baseline.</td>
<td>A</td>
</tr>
<tr>
<td>OBJECTS (baseline)</td>
<td>None</td>
<td>Specifies the name of a baseline for a baseline report.</td>
<td>BR</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>OBJINFO</td>
<td>None</td>
<td>Produces comments in the worklist that show the object identifiers and data set names for table spaces, tables, and indexes. An alter-type work ID ignores the OBJINFO keyword. A migrate-type work ID with a table space uses the OBJINFO keyword. However, a migrate-type work ID with a table ignores the OBJINFO keyword. This information can be useful when using the IBM DSN1COPY utility outside of ALTER or CHANGE MANAGER to move data. See the appropriate IBM documentation for information about using DSN1COPY to merge DB2 catalogs.</td>
<td>A</td>
</tr>
<tr>
<td>ORDERBY</td>
<td>None</td>
<td>Builds the ORDER BY clause when a clustering index is present on the table that is being unloaded. The use of this keyword can result in significant overhead. See also the NOORDERBY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>OVERRIDE( BASELINE-DELETE )</td>
<td>None</td>
<td>For a batch run type, deletes the baseline specified for a baseline profile. To delete the baseline, you must include the following keywords in the ALUIN DD in the batch JCL: ■ BLNAME a.b ■ BLPROFILE a.b ■ OVERRIDE( BASELINE-DELETE ) ■ NOBASELINE The name of the baseline must be fully qualified. The BLNAME and the BLPROFILE ALUIN keywords cannot contain a template.</td>
<td>B</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>OVERRIDE( CKSEGSIZE )</td>
<td>None</td>
<td>Shows the differences between a simple table space and a segmented table space defined with a SEGSIZE of 4.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( ENQFAILRC4 )</td>
<td>None</td>
<td>Produces a return code of 4 when the deletion of a baseline fails because the baseline is in use.</td>
<td>B</td>
</tr>
<tr>
<td>OVERRIDE( FKNN )</td>
<td>None</td>
<td>Shows the differences between two foreign keys when one foreign key has a three-part explicit name and the other foreign key has no name (part three is not specified).</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( GENOBID )</td>
<td>None</td>
<td>Specifies the value of the OBID as a comment for a CREATE TABLE statement in the CDL. CHANGE MANAGER uses this comment when you compare a file or DB2 catalog to either a catalog baseline or a DB2 catalog. The comment also appears in the baseline report.</td>
<td>B, BR, C</td>
</tr>
<tr>
<td>OVERRIDE( INCLUDEPARENTS )</td>
<td>None</td>
<td>Includes the parent objects of tables or table spaces in the scope of a catalog-to-catalog comparison when you are specifying the object types and object names for Compare1 and Compare2 to explicitly define the scope of the comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( IXPADDEDN )</td>
<td>None</td>
<td>Uses the default of PADDED NO for an index, regardless of the value in the DB2 subsystem parameter DSNZPARM.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( IXPADDEdy )</td>
<td>None</td>
<td>Uses the default of PADDED YES for an index, regardless of the value in the DB2 subsystem parameter DSNZPARM.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>-----------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>OVERRIDE( KEEP-ALL-PRIMARY)</td>
<td>UCPKPALL=Y</td>
<td>Includes the primary key constraint in a baseline or comparison when you exclude unique constraints.</td>
<td>B, C</td>
</tr>
<tr>
<td>OVERRIDE( LONGVARCOLS )</td>
<td>None</td>
<td>Shows the differences between two columns when one column has a LONG VARCHAR data type and the other column has a VARCHAR data type.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NO-PRIMARY-CONSTRAINTS)</td>
<td>UCPKPALL=N</td>
<td>Ignores the primary key constraint in a baseline or comparison when you exclude unique constraints.</td>
<td>B, C</td>
</tr>
<tr>
<td>OVERRIDE( NOAPPEND )</td>
<td>None</td>
<td>Ignores the APPEND attribute of a table in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOCLONES )</td>
<td>None</td>
<td>Ignores clone tables in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NODSSIZE )</td>
<td>None</td>
<td>Ignores the DSSIZE attribute of a table space in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOENFORCED )</td>
<td>None</td>
<td>Ignores the ENFORCED attribute of a foreign key in a table in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOGBPCACHE )</td>
<td>None</td>
<td>Ignores the GBPCACHE attribute for table spaces in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOINLINELOB )</td>
<td>None</td>
<td>Ignores the INLINE attribute for table spaces in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOIXCLOSE )</td>
<td>None</td>
<td>Ignores the CLOSE attribute of an index in comparison</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOIXCOMPRESS )</td>
<td>None</td>
<td>Ignores the COMPRESS attribute of an index in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOIXCOPY )</td>
<td>None</td>
<td>Ignores the COPY attribute of an index in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOIXPADDLED )</td>
<td>None</td>
<td>Ignores the PADDLED attribute of an index in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOLIMITKEYS )</td>
<td>None</td>
<td>Ignores LIMITKEY attribute of a partitioned table space in a comparison when you include partition attributes in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>OVERRIDE( NOLOG )</td>
<td>None</td>
<td>Ignores the LOG attribute of a table space in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOPICESIZE )</td>
<td>None</td>
<td>Ignores the PIECESIZE attribute of an index in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPACTIVE )</td>
<td>None</td>
<td>Ignores the active version attribute for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPASUTIME )</td>
<td>None</td>
<td>Ignores the ASUTIME option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPCCSID )</td>
<td>None</td>
<td>Ignores the CCSID option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSP_COLLID )</td>
<td>None</td>
<td>Ignores the package collection ID option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSP_COMMITRETURN )</td>
<td>None</td>
<td>Ignores the commit on return option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSP_CONACCRESO )</td>
<td>None</td>
<td>Ignores the concurrent access resolution option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSP_CURRENTDATA )</td>
<td>None</td>
<td>Ignores the CURRENTDATA option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSP_DATAACC )</td>
<td>None</td>
<td>Ignores the SQL DATA ACCESS option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSP_DATEFORMAT )</td>
<td>None</td>
<td>Ignores the date format option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSP_DBINFO )</td>
<td>None</td>
<td>Ignores the DBINFO option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSP_DEBUGMODE )</td>
<td>None</td>
<td>Ignores the DEBUG MODE option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSP_DECIMAL )</td>
<td>None</td>
<td>Ignores the DECIMAL option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>OVERRIDE( NOSPDEFERPREPARE )</td>
<td>None</td>
<td>Ignores the DEFER PREPARE option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPDEGREE )</td>
<td>None</td>
<td>Ignores the degree option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPDETERMINISTIC )</td>
<td>None</td>
<td>Ignores the deterministic option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPDYNRULES )</td>
<td>None</td>
<td>Ignores the dynamic rules option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPEXPLAIN )</td>
<td>None</td>
<td>Ignores the EXPLAIN option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPEXTNAME )</td>
<td>None</td>
<td>Ignores the external name option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPFORUPD )</td>
<td>None</td>
<td>Ignores the FOR UPDATE clause option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPIMMEDWRITE )</td>
<td>None</td>
<td>Ignores the IMMEDIATE WRITE option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPISOLEVEL )</td>
<td>None</td>
<td>Ignores the isolation level option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPJAVA )</td>
<td>None</td>
<td>Ignores the JAR SCHEMA, JAR ID, and JAVA SIGNATURE options for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPKEEPDYN )</td>
<td>None</td>
<td>Ignores the KEEP DYNAMIC option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPLANGUAGE )</td>
<td>None</td>
<td>Ignores the implementation language option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPMAXFAIL )</td>
<td>None</td>
<td>Ignores the maximum number of failures option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>OVERRIDE( NOSPOPTHINT )</td>
<td>None</td>
<td>Ignores the OPTHINT option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPACKOWNER )</td>
<td>None</td>
<td>Ignores the package owner option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPARAMETERS )</td>
<td>None</td>
<td>Ignores the parameter attributes for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPARMCCSID )</td>
<td>None</td>
<td>Ignores the PARAMETER CCSID option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPARMDATATYPE )</td>
<td>None</td>
<td>Ignores the data type of the parameter for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPARMFORBIT )</td>
<td>None</td>
<td>Ignores the subtype of the parameter for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPARMINOUT )</td>
<td>None</td>
<td>Ignores the row type of the parameter for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPARMLOC )</td>
<td>None</td>
<td>Ignores the location of the parameter for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPARMNAME )</td>
<td>None</td>
<td>Ignores the name of the parameter for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPARMORDER )</td>
<td>None</td>
<td>Ignores the ordinal number of the parameter for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPARMSTYLE )</td>
<td>None</td>
<td>Ignores the PARAMETER STYLE option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPARMVCHAR )</td>
<td>None</td>
<td>Ignores the varying length string parameter for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPPROGRAMTYPE )</td>
<td>None</td>
<td>Ignores the PROGRAM TYPE option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>OVERRIDE( NOSPQUALIFIER )</td>
<td>None</td>
<td>Ignores the qualifier option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPREGISTERS )</td>
<td>None</td>
<td>Ignores the SPECIAL REGISTER option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPRELEASEAT )</td>
<td>None</td>
<td>Ignores the release option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPREOPT )</td>
<td>None</td>
<td>Ignores the REOPT option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPRESULTSET )</td>
<td>None</td>
<td>Ignores the result sets option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPROUNDING )</td>
<td>None</td>
<td>Ignores the ROUNDDING option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPRUNOPTS )</td>
<td>None</td>
<td>Ignores the RUNOPTS option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPSECURITY )</td>
<td>None</td>
<td>Ignores the EXTERNAL SECURITY option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPSQLPATH )</td>
<td>None</td>
<td>Ignores the SQL path (PATHSCHEMAS) option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE(NOSPSTAYRESIDENT)</td>
<td>None</td>
<td>Ignores the STAY RESIDENT option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPTEXT )</td>
<td>None</td>
<td>Ignores the text option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPTFORMAT )</td>
<td>None</td>
<td>Ignores the time format option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPVALIDATE )</td>
<td>None</td>
<td>Ignores the VALIDATE option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>OVERRIDE( NOSPWLMEENV )</td>
<td>None</td>
<td>Ignores the WLM environment option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSPWLNMEST )</td>
<td>None</td>
<td>Ignores the WLM environment for nested calls option for stored procedures in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSTOREDPROCS )</td>
<td>None</td>
<td>Ignores stored procedures in the scope of a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOSUBTYPE )</td>
<td>None</td>
<td>Ignores the subtype of a column of a table in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOTBAUDIT )</td>
<td>None</td>
<td>Ignores the AUDIT attribute of a table in a comparison</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOTBCOLORER )</td>
<td>None</td>
<td>Ignores differences in the order of the columns in a table during a comparison. If you specify OVERRIDE( NOTBCOLORER ) and you add a new column to the table, the product adds the column to the end of the table.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOTBDEFSIZE )</td>
<td>None</td>
<td>Ignores the default value of a column in a table in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOTBTSAUTO )</td>
<td>None</td>
<td>Does not apply automatic change rules to match up all of the table’s table space names when comparing at the table level.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOTSCLOSE )</td>
<td>None</td>
<td>Ignores the CLOSE attribute of a table space in a comparison</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( NOVWQUAL )</td>
<td>None</td>
<td>Ignores the QUALIFIED attribute (explicit qualifier) of a view in a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>OVERRIDE( SPOWNER )</td>
<td>None</td>
<td>Shows the differences between the owners of stored procedures. By default, the product does not compare the owners.</td>
<td>B, BR, C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------</td>
<td>-------------</td>
<td>----------------------------------</td>
</tr>
</tbody>
</table>
| OVERRIDE(SPTTEXT-TRACE) | None | Provides a hex dump of the text for a stored procedure when all of the following conditions are met:  
- The product retrieves the information from the SYSIBM catalog.  
- Compare detects a difference between the text for Compare1 and Compare2. | B, BR, C |
| OVERRIDE(UCNN) | None | Shows the differences between two unique constraints when one unique constraint has a three-part explicit name and the other unique constraint has no name (part three is not specified). | C |
| PARALLEL | PARALLEL=Y | For the Database Administration and BMC Object Administration for DB2 solutions, creates commands that Execution can use to run a worklist in parallel.  
To run a worklist in parallel, the PARALLEL keyword must be specified in the ALUIN input stream, and the ACM_PARALLEL_WORKLST keyword in the AJXPOFIN input stream must be set to Y or the PARALLEL YES keyword must be added to the AEXPIN input stream.  
See also the NOPARALLEL keyword in this table. | A |
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARTCOPY</td>
<td>PARTCPY=Y</td>
<td>Uses the LOADPLUS, NGT Copy, NGT Recover, or REORG PLUS utility to create a partition-level image copy of a partitioned table space or index. NGT Reorg creates all partition-level copies with dynamic allocation. NGT Reorg disregards this keyword. See also the NOPARTCOPY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>PIC</td>
<td>PIC=[Y,N]</td>
<td>Takes an image copy of each table space before it drops a database or table, or before it drops and reorganizes a table space.</td>
<td>A</td>
</tr>
<tr>
<td>PKEYPROP</td>
<td>None</td>
<td>Propagates column attribute changes on a primary key to the tables of the foreign keys that are connected to that primary key. This propagation applies only to changes in the DATATYPE, LENGTH, DECIMAL, FIELDPROC, and FIELDPROCPARM attributes.</td>
<td>A</td>
</tr>
<tr>
<td>PLANMGMTBASIC</td>
<td>PLANMGMGT=B</td>
<td>Specifies BASIC as the value for PLANMGMGT when a REBIND statement is executed.</td>
<td>A</td>
</tr>
<tr>
<td>PLANMGMTEXTENDED</td>
<td>PLANMGMGT=E</td>
<td>Specifies EXTENDED as the value for PLANMGMGT when a REBIND statement is executed.</td>
<td>A</td>
</tr>
<tr>
<td>PROTECT</td>
<td>None</td>
<td>Protects the baseline that you are creating from being deleted.</td>
<td>A, B</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>REBINDADVANCED</td>
<td>REGENATR=N</td>
<td>Rebinds an advanced trigger by generating the REBIND PACKAGE command in a worklist. You must manually insert this keyword into the ALUIN input stream. Note: For advanced triggers, you must specify REBIND PACKAGE, not REBIND TRIGGER PACKAGE. See also the REGENADVANCED keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>REBINDNATIVE</td>
<td>REGENNSP=N</td>
<td>Rebinds a package that is associated with a native SQL stored procedure by generating the REBIND PACKAGE command in a worklist. You must manually insert this keyword into the ALUIN input stream. See also the REGENNATIVE keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>REBUILD</td>
<td>None</td>
<td>Creates a new worklist if a previous worklist was created for the same work ID. This keyword is required if the status of the work ID is analyzed.</td>
<td>A</td>
</tr>
<tr>
<td>REFRESH</td>
<td>REFRESH=Y</td>
<td>Uses the REFRESH TABLE statement to refresh the data in a materialized query table (MQT). You can manually add this keyword to the ALUIN input stream. See also the NOREFRESH keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>REGENADVANCED</td>
<td>REGENATR=Y</td>
<td>Rebinds an advanced trigger by generating the ALTER TRIGGER with a REGENERATE VERSION clause in a worklist. You must manually insert this keyword into the ALUIN input stream. See also the REBINDADVANCED keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>REGENIDENTITY</td>
<td>None</td>
<td>Enables DB2 to assign a unique value to a column that is defined with GENERATED ALWAYS AS IDENTITY. You must manually add this keyword to the ALUIN input stream. See also the NOREGENIDENTITY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>REGENNATIVE</td>
<td>REGENNSP=Y</td>
<td>Rebinds a package that is associated with a native SQL stored procedure by generating the ALTER PROCEDURE statement with the REGENERATE clause in a worklist. You must manually insert this keyword into the ALUIN input stream. See also the REBINDNATIVE keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>REGISTER (parameters)</td>
<td>COPYDD01=R</td>
<td>Registers an inline copy of a data set for the LOADPLUS, REORG PLUS, and NGT Recover utilities. Local-copy parameters (parameters) for the REGISTER keyword are COPY01 and COPY02. Separate the parameters with commas or blanks. Remote-copy parameters (parameters) for the REGISTER keyword are RECV01 and RECV02. Separate the parameters with commas or blanks.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>REM ('comment')</td>
<td>None</td>
<td>Allows a baseline comment (with a length of 62 characters) to be entered.</td>
<td>B</td>
</tr>
<tr>
<td>REORGALL</td>
<td>REORGALL=Y</td>
<td>Reorganizes a table space when a reorganization would be applicable. See also the REORGPENDONLY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>REORGREF</td>
<td>REORGREF=Y</td>
<td>Reorganizes a table space by using the SHRLEVEL REFERENCE option when an online reorganization (SHRLEVEL CHANGE) is not applicable or is not requested. Note: NGT Reorg always performs an online reorganization (SHRLEVEL CHANGE) and disregards this keyword. See also the REORGONLINE keyword in this table.</td>
<td>A</td>
</tr>
</tbody>
</table>
| REORGONLINE     | REORGONL=Y                        | Reorganizes a table space by using an online reorg (SHRLEVEL CHANGE). If an online reorg is not applicable, the product can reorganize the table space by using the SHRLEVEL REFERENCE or SHRLEVEL NONE option. Either select one of these options on the Analysis Reorg Option panel, or add the relevant keyword to the ALUIN input stream: 
  - REORGREF (for SHRLEVEL REFERENCE) 
  - REORGNONE (for SHRLEVEL NONE) 
Note: NGT Reorg always performs an online reorganization (SHRLEVEL CHANGE) and disregards this keyword. See also the NOREORGONLINE keyword in this table. | A                                |
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORGNONE</td>
<td>REORGREF=N</td>
<td>Reorganizes a table space by using the SHRLEVEL NONE option when an online reorg (SHRLEVEL CHANGE) is not applicable or is not requested. See also the REORGREF keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>REORGPENDONLY</td>
<td>REORGALL=N</td>
<td>Reorganizes a table space when the table space requires a reorganization (for example, the table space is in REORG-pending status). See also the REORGALL keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>REPLACE</td>
<td>None</td>
<td>Specifies that existing work IDs are replaced with the REPLICATE WORKID statement. See also the NOREPLACE keyword in this table.</td>
<td>P</td>
</tr>
<tr>
<td>REPLACEWORKID a.b</td>
<td>None</td>
<td>Replaces an existing work ID. The AEXIN input stream also uses this keyword.</td>
<td>E, F, I, P</td>
</tr>
<tr>
<td>REPORT (DETAIL)</td>
<td>None</td>
<td>Creates comparison report information as comments in the CDL file. The comments include all of the CREATE, ALTER, and DROP statements for a comparison.</td>
<td>C</td>
</tr>
<tr>
<td>REPORTTYPE BASELINE</td>
<td>None</td>
<td>Specifies the baseline report.</td>
<td>BR</td>
</tr>
<tr>
<td>RESMAXIC nnnnn</td>
<td>None</td>
<td>Determines which image copy data set holds the data that is to be migrated by using the NGT Recover product. The variable nnnnn indicates the number of image copies back from the current copy. For example, if nnnnn is -5, the parameter uses the data set for the image copy that was taken 5 copies ago. The valid range for nnnnn is 0 through -9999.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>RETAINMAX $n$</td>
<td>None</td>
<td>Retains $n$ number of complete baselines and delete all others that were created with the specified baseline profile. See also the RETAINMAX $nnnn$ keyword in the AJXIN input stream.</td>
<td>B</td>
</tr>
<tr>
<td>REUSEWORKID</td>
<td>None</td>
<td>Replaces the work ID in the ALUIN input stream with the work ID listed in the previous -WKID worklist command.</td>
<td>P</td>
</tr>
<tr>
<td>ROLEOWN</td>
<td>ROLEOWN=Y</td>
<td>Allows objects to be created where the owner, creator, or grantor is a ROLE.</td>
<td>A</td>
</tr>
<tr>
<td>RPLUSYNCAUTO</td>
<td>RPLUSYNC=A</td>
<td>For the Copy Migration feature, includes only the table spaces that have changed (since the last migration of the table spaces) in the migration file that the NGT Recover IMPORT command imports. The Copy Migration feature is included in the following solutions:</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Recovery Management</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- BMC Recovery for DB2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- BMC Object Administration for DB2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- BMC Next Generation Technology Database Administration for DB2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Database Administration</td>
<td></td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------</td>
<td>-------------</td>
<td>----------------------------------</td>
</tr>
</tbody>
</table>
| RPLUSYNCREPLACE | RPLUSYNC=R | For the Copy Migration feature, includes all table spaces in the migration file that the NGT Recover IMPORT command imports. The Copy Migration feature is included in the following solutions:  
- Recovery Management  
- BMC Recovery for DB2  
- BMC Object Administration for DB2  
- BMC Next Generation Technology Database Administration for DB2  
- Database Administration | A |
<p>| SCOPE1 <em>objectOwner., objectName</em> | None | Names a baseline profile or an outbound migrate profile that contains scope rules. You would use SCOPE1 <em>SSID.objectOwner.objectName</em> if you wanted to name a baseline profile on a remote subsystem. To specify a three-part name, you would have edit the keyword manually. SCOPE1 corresponds to the primary set of input in a comparison. This keyword replaces the SCOPEPROFILE keyword. | C |</p>
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCOPE2 <code>objectOwner.objectName</code></td>
<td>None</td>
<td>Names a baseline profile or an outbound migrate profile that contains scope rules. SCOPE2 <code>SSID.objectOwner.objectName</code> names a baseline profile on a remote subsystem. To specify a three-part name, you must manually edit the keyword in the ALUIN input stream. SCOPE2 corresponds to the secondary set of input in a comparison. This keyword replaces the SCOPEPROFILE keyword.</td>
<td>C</td>
</tr>
<tr>
<td>SCOPERULE1 <code>(objectOwner object name_part1. name_part2)</code></td>
<td>None</td>
<td>Specifies the owner and the name of an object that limits the scope of database and table space comparisons when comparing two DB2 catalogs. Compare uses SCOPERULE1 when you specify one catalog object and its dependents as the scope of the comparison. SCOPERULE1 corresponds to the primary set of input in the comparison. A space must separate each parameter. Note: The part1 and part2 names are optional.</td>
<td>C</td>
</tr>
<tr>
<td>SCOPERULE2 <code>(objectOwner object [name_part1. name_part2])</code></td>
<td>None</td>
<td>Specifies the owner and the name of an object that limits the scope of database and table space comparisons when comparing two DB2 catalogs. Compare uses SCOPERULE2 when you specify one catalog object and its dependents as the scope of the comparison. SCOPERULE2 corresponds to the secondary set of input in the comparison. A space must separate each parameter.</td>
<td>C</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>SCOPETYPE <code>type,type,type</code></td>
<td>None</td>
<td>Specifies the type of scope for a comparison. The following values are valid for type:</td>
<td>C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ BASELINE is valid when specified in a profile.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ DDL is valid when specified for objects that are contained in a data definition language (DDL) file for the scope.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ MIGRATE is valid when specified in a profile.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ PROFILES is valid when you use a catalog baseline profile or an outbound migrate profile as the source of the scope rules in a comparison of one DB2 catalog to another DB2 catalog.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ RULE is valid for comparing one DB2 catalog to another DB2 catalog when you specify a catalog object and its dependents as the scope of the comparison.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ WORKLIST is valid when specified for a comparison that uses the objects that are contained in a worklist for the scope.</td>
<td></td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>SHOWRULES</td>
<td>None</td>
<td>Displays the automatic change rules that are created when either of the following conditions exist:</td>
<td>C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ An outbound migrate profile (indicated by the CHANGERULESIN2 keyword) is not specified for a comparison of one DB2 catalog to another DB2 catalog.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ An outbound migrate profile is specified for any comparison (including a comparison of one DB2 catalog to another DB2 catalog).</td>
<td></td>
</tr>
<tr>
<td>SHRLEVELCHG</td>
<td>None</td>
<td>Uses image copies that are marked as either SHRLEVEL REFERENCE or SHRLEVEL CHANGE.</td>
<td>A</td>
</tr>
<tr>
<td>SHRLEVELREF</td>
<td>None</td>
<td>Uses image copies that are marked as SHRLEVEL REFERENCE.</td>
<td>A</td>
</tr>
<tr>
<td>SINGLEPHASE</td>
<td>None</td>
<td>Creates a single-phase worklist for migrating data structures within the same DB2 subsystem.</td>
<td>A</td>
</tr>
<tr>
<td>SMSINCLUDE [parameter]</td>
<td>None</td>
<td>Identifies the IBM Storage Management Subsystem (SMS) parameters that are included on the -AMS command in the worklist.</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The following values are valid for parameter:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ STORCLAS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ MGMTCLAS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ DATACLAS</td>
<td></td>
</tr>
<tr>
<td>SORTDEVT</td>
<td>None</td>
<td>Specifies the device type to use for dynamically allocated SORTWORK data sets.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
<td></td>
</tr>
<tr>
<td>SOURCETYPE</td>
<td>Defines the type of input to the Import component or the type of baseline to the Baseline component. The following values are valid for parameter for Import in CHANGE MANAGER: CDL, DDL, WORKLIST. The following values are valid for parameter for Baseline: DDL and CATALOG. DDL is a valid value for parameter for Import in ALTER.</td>
<td>B, I</td>
<td></td>
</tr>
<tr>
<td>SPANNEDRECS</td>
<td>Uses spanned records to unload and load XML or LOB columns. DYNULD = Y is required when you unload and load XML or LOB columns by using SPANNEDRECS. Note: You cannot use SPANNEDRECS with the following ALUIN keywords: NOLOBDATA, NOXMLDATA, NOUTILCOPY, BMCFASTLOAD</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>SPTEXT( EXACT )</td>
<td>Compares each character in the text for native SQL stored procedures verbatim. Compare ignores the X'0D' carriage return.</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>SPTEXT( NOEXACT )</td>
<td>Ignores extra blanks, blank lines, and character case (uppercase and lowercase) when comparing the text for native SQL stored procedures.</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------</td>
<td>-------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>SSID ssid</td>
<td>None</td>
<td>Identifies the DB2 subsystem ID or the DB2 data sharing group attachment name. The SSID keyword must match the -SSID command in the worklist. This parameter is required. If the SSID is specified in the JCL in an EXEC statement in a cataloged procedure (which begins with a PROC statement), its value overrides the value of the SSID keyword. The AEXIN and AJXIN input streams also use this keyword.</td>
<td>A, B, BR, E, F, I, J, P, S</td>
</tr>
<tr>
<td>STOPCOMMAND</td>
<td>STOPCOMMAND=Y</td>
<td>Generates an AT(COMMIT) statement for every STOP command in the worklist. This keyword can be manually added in the ALUIN input stream. See also the NOSTOPCOMMAND keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>STOPICP</td>
<td>STOPICP=Y DUAL=Y</td>
<td>Specifies that all new partitioned objects must use table-controlled partitioning. This keyword overrides the STOPICP=N or DUAL=N installation options. This keyword must be manually added in the ALUIN input stream. See also the ALLOWICP keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>STOPLIST <code>creatorName</code></td>
<td>STOPLIST=<code>creatorName</code></td>
<td>Specifies the eight-character creator name for a migrate-type work ID that identifies a list of tables that should not be analyzed (stop list). An example of a table that should not be analyzed might be those that are defined with the DATA CAPTURE CHANGES attribute. If a table is referenced in a work ID that is included in the stop list, Analysis issues an error message. The default value is blank. See also the NOSTOPLIST keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>SYNCPOINT <code>parameter</code></td>
<td>SYNCPNT <code>parameter</code></td>
<td>Creates additional -SYNC commands in a worklist, based on the number of -SQL commands since the last -SYNC command. The variable <code>parameter</code> specifies the maximum number of -SQL commands that can be in the worklist before a -SYNC command is created. Valid values for <code>parameter</code> are from 0 through 99. The default is 10. If this value is reached, Analysis places an additional -SYNC command before the next -SQL command. Any -SYNC command in the worklist resets the count of -SQL commands to zero. The -SYNC commands that this keyword generates are in addition to the -SYNC commands that Analysis automatically generates.</td>
<td>A</td>
</tr>
<tr>
<td>TABLEACCESS</td>
<td>TABLEACC=Y</td>
<td>Allows outside access to tables during worklist execution. You must manually add this keyword to the ALUIN input stream.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>TABLEALL</td>
<td>TABLEALL=Y</td>
<td>Enables the BMCSTATS or IBM RUNSTATS utility to gather column statistics on all of the columns in all of the tables in a table space by including the TABLE ALL parameter in the -BMCS or -RNST worklist command.</td>
<td>A</td>
</tr>
</tbody>
</table>
| TCPROTATE            | None                              | Creates an ALTER TABLE ROTATE PARTITION statement in the worklist when you specify a value for the Rotate Parts field on the Tablespace Parts List panel for the work ID.  
See also the NOTCPROTATE keyword in this table.  
**WARNING:** When the product executes the ALTER TABLE ROTATE PARTITION statement, DB2 deletes all of the data in the rotated partitions of the table. | A                               |
| TERMINATOR character | BLRTERMC=#                        | Specifies the character that the baseline report uses to separate SQL statements.  
The default value is a pound sign (#). The value of character can be any character except a blank, comma (,), double quotation mark ("), single quotation mark (‘), left parenthesis ([), right parenthesis (]), or underscore (_).  
You must manually add this keyword to the ALUIN input stream. | BR                              |
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Corresponding installation option</th>
<th>Description</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>THRESHOLD $n$</td>
<td>None</td>
<td>Specifies the integer that limits the number of objects that the components pull from the DB2 catalog by using the nonunique catalog indexes. The default limit is 100,069 objects. The valid range of values is 1 through 2,147,483,646 (or $2^{31}-2$). If the number of objects that are requested exceeds the limit, the nonunique catalog indexes are ignored. The relevant table space in the catalog is then scanned for the requested objects. For example, if the THRESHOLD is set to 100,069, and you request 100,100 objects, the product fully scans the table space instead of opening 100,100 DB2 cursors.</td>
<td>B, C</td>
</tr>
<tr>
<td>TRANSFORM</td>
<td>None</td>
<td>Checks the text for a view, trigger, check constraint, or materialized query table (MQT). Analysis then extracts the required information that is needed to set the authorizations correctly. You must manually add the keyword to the ALUIN input stream.</td>
<td>A</td>
</tr>
<tr>
<td>TRIAL</td>
<td>None</td>
<td>Specifies that the ALUIN input stream contains CM/PILOT Data Manipulation Language (DML), but that Analysis does not generate a worklist. This keyword can also specify that Analysis generates only a report when the DELETE WORKID, DELETE SYNCTABLE, or REPLICATE WORKID statement is used.</td>
<td>A, P</td>
</tr>
<tr>
<td>TVALOFF</td>
<td>None</td>
<td>Does not use text validation for triggers. The VALOFF ALUIN keyword replaces this keyword.</td>
<td>A, C, I</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>TVALWARN</td>
<td>None</td>
<td>Treats error conditions in trigger text as warnings. If the component finds an error, it processes the trigger without propagation. Execution fails if the errors in a worklist have not been corrected. For Compare and Import, you must manually add the keyword to the ALUIN input stream. The VALWARN ALUIN keyword replaces this keyword.</td>
<td>A, C, I</td>
</tr>
<tr>
<td>UNLDCOLL</td>
<td>UNLDCOLL= Y</td>
<td>Requires explicit column lists on all of the UNLOAD PLUS utility unloads. To provide explicit column lists on the IBM UNLOAD utility unloads, you must manually insert the keyword in the ALUIN input stream.</td>
<td>A, J</td>
</tr>
<tr>
<td>UNLOADCOPY-xx</td>
<td>None</td>
<td>Specifies that an unload occurs from an image copy. Execution determines which particular image copy to use. Analysis generates INFILE IMAGECOPY FULL xx. For more information, see the UNLOAD PLUS for DB2 Reference Manual.</td>
<td>A</td>
</tr>
<tr>
<td>UNLOADCOPYDS-xx</td>
<td>None</td>
<td>Specifies that an unload occurs from an image copy. Analysis determines the particular image copy to use. Analysis generates INFILE IMAGECOPY DDName. For more information, see the UNLOAD PLUS for DB2 Reference Manual.</td>
<td>A</td>
</tr>
<tr>
<td>UNLOADEMPTY</td>
<td>UNLDEMT= Y</td>
<td>Unloads tables that the IBM RUNSTATS utility indicates are empty. See also the NOUNLOADEMPTY keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>UNLOADTABLE</td>
<td>None</td>
<td>Specifies that an unload occurs from the DB2 table data set instead of from an image copy.</td>
<td>A</td>
</tr>
<tr>
<td>USENGTR</td>
<td>USENGTR=Y</td>
<td>Specifies that data reorganization is performed by the NGT Reorg utility.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>See also the NOUSENGTR keyword in this table.</td>
<td></td>
</tr>
<tr>
<td>UTILCOPY</td>
<td>UTILCOPY=Y</td>
<td>Takes an image copy with the IBM REORG or LOAD utility or with the REORG PLUS, LOADPLUS, or NGT Recover utility. If you specify UTILCOPY and DYNCOPY, the utilities will not build the image copies. Only the NGT Copy or NGT Recover utility or the IBM COPY utility can build image copies when you select DYNCOPY. See also the NOUTILCOPY keyword in this table. NGT Reorg always creates inline copies with dynamic allocation. NGT Reorg disregards this keyword.</td>
<td>A</td>
</tr>
<tr>
<td>UTILSTATS</td>
<td>STATS=U</td>
<td>Uses the IBM REORG or LOAD utility whenever possible to collect statistics. NGT Reorg disregards this ALUIN keyword.</td>
<td>A, F</td>
</tr>
<tr>
<td>VALOFF</td>
<td>None</td>
<td>Does not use text validation for triggers, views, or materialized query tables (MQTs). This keyword replaces the TVALOFF and VVALOFF ALUIN keywords.</td>
<td>A, C, I</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>VALWARN</td>
<td>None</td>
<td>Treats error conditions in trigger, view, materialized query table (MQT), or index text as warnings. If the component finds an error, the component processes the text without propagation. Execution fails if the errors in a worklist have not been corrected. For Compare and Import, you must manually add the keyword to the ALUIN input stream. This keyword replaces the TVALWARN and VVALWARN ALUIN keywords.</td>
<td>A, C, I</td>
</tr>
<tr>
<td>VVALloff</td>
<td>None</td>
<td>Does not use text validation for views. You must verify authorizations for any unqualified dependent objects. The VALOFF ALUIN keyword replaces this keyword.</td>
<td>A, C, I</td>
</tr>
<tr>
<td>VVALPROP</td>
<td>VVALPROP=Y</td>
<td>Specifies that extended view text propagation is supported by ALTER or CHANGE MANAGER. See also the NOVVALPROP keyword in this table.</td>
<td>A</td>
</tr>
<tr>
<td>VVALWARN</td>
<td>None</td>
<td>Downgrades conditions that are not found and certain syntax errors to warnings. If an error is found, the diagnostic log records the specific error messages and the component does not propagate changes to the view text. Execution fails if the errors in a worklist have not been corrected. For Compare and Import, you must manually add the keyword to the ALUIN input stream. The VALWARN ALUIN keyword replaces this keyword.</td>
<td>A, C, I</td>
</tr>
<tr>
<td>Keyword</td>
<td>Corresponding installation option</td>
<td>Description</td>
<td>Used by CHANGE MANAGER component</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
</tbody>
</table>
| WLORDER order   | None                              | Tells Analysis how to sort objects in a worklist. The following values are valid for `order`:
- CARD sorts the worklist by each table’s cardinality in a descending sequence.
- NAME sorts the worklist by table order in ascending sequence, according to the table owner and table name.
- AUTO sorts the worklist by either table cardinality or table order, depending on whether the worklist is processed in parallel.
  If the Database Administration or BMC Object Administration for DB2 solution processes the worklist in parallel, Analysis sorts the worklist by table cardinality. Otherwise, it sorts the worklist by table order.
- NONE generates the objects in the worklist in an unsorted, random order. (See also the NOWLORDER keyword in this table.) | A                                |
| WLORDERMSG      | None                              | Records in the SYSPRINT data set and in the worklist the amount of time to sort a worklist.                                                                                                                                                                                                                                                  | A                                |
| WORKID a.b      | None                              | Specifies the work ID to use. Execution fails if this work ID does not match the work ID that the -WKID command in the worklist specifies. The AEXIN and AJXIN input streams also use this keyword.                                                                                                                                                 | A, B, E, I                       |
The default operating environment is controlled by a number of option values that are defined in the installation options module. You can override some of these installation options with a keyword in the ALUIN, AJXIN, or AEXIN input stream.

Table 17 on page 139 lists each ALTER and CHANGE MANAGER installation option, the overriding keyword, and the input stream in which the keyword appears. ALTER and CHANGE MANAGER use the product options file (POF), which contains keywords and values for ALTER, CHANGE MANAGER, and JCL Generation options. These POF keywords replaced some of the installation options that were used for JCL Generation and the product options. Table 17 on page 139 also lists the POF keywords that replaced the installation options.

For a description of and the values for the installation options, see “ALTER and CHANGE MANAGER installation options” on page 159. For a description of and the values for the POF options, see “JCL Generation product options” on page 181. For a description of the keywords, see “AEXIN keywords” on page 55 through “ALUIN keywords” on page 71.

### Table 17: Installation options

<table>
<thead>
<tr>
<th>Installation option</th>
<th>Keyword</th>
<th>Type of keyword</th>
<th>Overrides or replaces</th>
<th>Used by CHANGE MANAGER component</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLOC</td>
<td>None</td>
<td>None</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>AMS</td>
<td>ACM_AMS</td>
<td>POF</td>
<td>Replaces</td>
<td></td>
</tr>
<tr>
<td>ANP</td>
<td>None</td>
<td>None</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>AUTHSW</td>
<td>AUTHSW</td>
<td>ALUIN</td>
<td>Overrides</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AUTHSWGLID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AUTHSWOFF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>NOAUTHSW</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BASDIAG</td>
<td>ACM_BASDIAG</td>
<td>POF</td>
<td>Replaces</td>
<td></td>
</tr>
<tr>
<td>BASE</td>
<td>None</td>
<td>None</td>
<td>NA</td>
<td></td>
</tr>
</tbody>
</table>
## Keywords that override installation options

<table>
<thead>
<tr>
<th>Installation option</th>
<th>Keyword</th>
<th>Type of keyword</th>
<th>Overrides or replaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>BINDONIX</td>
<td>BINDONIX</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>NOBINDONIX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLRTERMC</td>
<td>TERMINATOR</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>BMCCHECK</td>
<td>BMCCHECK</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>IBMCHECK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMCCOPY</td>
<td>BMCCOPY</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>IBMCOPY</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>NOCOPY</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>NOCOPYFORCE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMCFASTL</td>
<td>BMCFASTLOAD</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>NOBMCFASTLOAD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMCLOAD</td>
<td>BMCLOAD</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>IBMLOAD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMCUUNLD</td>
<td>BMCUUNLOAD</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>BPOOLIX</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>BPOOLTS</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>BRPTDIAG</td>
<td>ACM_BRPFTDIAG</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>BRPTDSN</td>
<td>ACM_BRPFTDSN</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>CATAUDIT</td>
<td>CATAUDIT</td>
<td>AEXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>CATRECOV</td>
<td>CATRECOVER</td>
<td>AEXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>CCSID</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>CDLDSN</td>
<td>ACM_CDLDSN</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>CDLPS</td>
<td>ACM_CDLPS</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>CDLSS</td>
<td>ACM_CDLSS</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>CDLU</td>
<td>ACM_CDLU</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>CLONDATA</td>
<td>CLONEDATA</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>NOCLONEDATA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMP</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>CMPDIAG</td>
<td>ACM_CMPDIAG</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>CMPLIMIT1</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>CMPLIMIT2</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>CMPSCWBF</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>CMPSCWBS</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
</tbody>
</table>
### Keywords that override installation options

<table>
<thead>
<tr>
<th>Installation option</th>
<th>Keyword</th>
<th>Type of keyword</th>
<th>Overrides or replaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMPSPTXT</td>
<td>SPTEXT( EXACT )</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>SPTEXT( NOEXACT )</td>
<td>ALUIN</td>
<td></td>
</tr>
<tr>
<td>COPYDD01</td>
<td>COPY01</td>
<td>AJXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>COPYDDN REGISTER</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>COPYDD02</td>
<td>COPY02</td>
<td>AJXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>COPYDDN REGISTER</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>CPLDIAG</td>
<td>ACM_CPLDIAG</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>CPLWDSN</td>
<td>ACM_CPLWDSN</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>DASDMAN</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>DATE</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>DBRM1</td>
<td>ACM_DBRM1</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td></td>
<td>DBRM1</td>
<td>AJXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>DBRM2</td>
<td>ACM_DBRM2</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td></td>
<td>DBRM2</td>
<td>AJXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>DBRM3</td>
<td>ACM_DBRM3</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td></td>
<td>DBRM3</td>
<td>AJXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>DBRMLIB</td>
<td>DBRMLIB</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>DEFERUIX</td>
<td>DEFERUNIQUEIX</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>DISCARDS</td>
<td>DISCARDS</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>DUAL (ALTER and CHANGE MANAGER version 10.1 or 11.1 only)</td>
<td>ALLOWICP STOPICP</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>DYNCOPY</td>
<td>DYNCOPY NODYNCOPY</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>DYNUNLD</td>
<td>DYNUNLD NODYNUNLD</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>EAP</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>EIP</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>ENVP</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>EPP</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>Installation option</td>
<td>Keyword</td>
<td>Type of keyword</td>
<td>Overrides or replaces</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------</td>
<td>-----------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>EURO</td>
<td>EURO</td>
<td>AJXIN ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>EXCEPT</td>
<td>CKEXCEPTIONS</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>FEP</td>
<td>None</td>
<td>None NA</td>
<td></td>
</tr>
<tr>
<td>GENOBID</td>
<td>GENOBID NOGENOBID OVERRIDE(GENOBID)</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>GLID</td>
<td>ACM_GLID</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td></td>
<td>GLAUTHID</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>HSMVOL</td>
<td>HSMVOL NOHSMVOL</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>IMP</td>
<td>None</td>
<td>None NA</td>
<td></td>
</tr>
<tr>
<td>IMPDIAG</td>
<td>ACM_IMPDIA</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>JDSN</td>
<td>ACM_JDSN</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>JDSNB</td>
<td>ACM_JDSNB</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>JDSNighb</td>
<td>ACM_JDSNGB</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>JDSNBrib</td>
<td>ACM_JDSNBR</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>JDSNC</td>
<td>ACM_JDSNC</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>JDSNCpl</td>
<td>ACM_JDSNCPL</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>JDSNE</td>
<td>ACM_JDSNE</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>JDSNI</td>
<td>ACM_JDSNI</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>KEYCARD</td>
<td>KEYCARD</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>NOKEYCARD</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>LKVALID</td>
<td>IGNORELKERROR NOORDERLK</td>
<td>ALUIN</td>
<td>NA</td>
</tr>
<tr>
<td>LOCATION</td>
<td>None</td>
<td>None NA</td>
<td></td>
</tr>
<tr>
<td>LOCK</td>
<td>LOCK</td>
<td>AEXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>JOBLOCK</td>
<td>AJXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>LOG</td>
<td>LOG</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>MAXSYSUT</td>
<td>MAXSYSUT</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>MGMTCLAS</td>
<td>None</td>
<td>None NA</td>
<td></td>
</tr>
<tr>
<td>Installation option</td>
<td>Keyword</td>
<td>Type of keyword</td>
<td>Overrides or replaces</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------</td>
<td>-----------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>PLANMGMT=N</td>
<td>NOPLANMGMT, PLANMGMTBASIC, PLANMGMTEXTENDED</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>PARALLEL</td>
<td>NOPARALLEL, PARALLEL</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>PARTCPY</td>
<td>NOPARTCOPY, PARTCOPY</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>PC</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>PIC</td>
<td>ACM_PIC</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td></td>
<td>PIC</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>POFDS</td>
<td>POFDS</td>
<td>AJXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>PRODUCT</td>
<td>PRODUCT</td>
<td>AJXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>REGENATR</td>
<td>REBINDADVANCED, REGENADVANCED</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>REBLD</td>
<td>BMCREBUILD, IBMREBUILD, NOREBUILD</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>RECVDD01</td>
<td>RECV01</td>
<td>AJXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>COPYDDN, REGISTER</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>RECVDD02</td>
<td>RECV02</td>
<td>AJXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>COPYDDN, REGISTER</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>REFRESH</td>
<td>REFRESH, NOREFRESH</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>REGENNSP</td>
<td>REBINDNATIVE, REGENNATIVE</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>REORG</td>
<td>BMCREORG, IBMREORG</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>REORGALL</td>
<td>REORGPENDONLY, REORGALL</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>REORGONL</td>
<td>NOREORGONLINE, REORGONLINE</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>REORGREF</td>
<td>REORGREF, REORGNONE</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>Installation option</td>
<td>Keyword</td>
<td>Type of keyword</td>
<td>Overrides or replaces</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------------------------</td>
<td>-----------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>ROLEOWN</td>
<td>ROLEOWN</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>NOROLEOWN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPLUSYNC</td>
<td>RPLUSYNCAUTO</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>RPLUSYNCREPLACE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPTPL</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>SDSN</td>
<td>ACM_SDSN</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td></td>
<td>SDSN</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>SDSNE</td>
<td>ACM_SDSNE</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td></td>
<td>SDSNE</td>
<td>AJXIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>SEQI</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>SPANNED</td>
<td>NOSPANNEDRECS</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>SPANNEDRECS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPP</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>STATHIST</td>
<td>HISTORYALL</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>NOHISTORY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STATS</td>
<td>UTILSTATS</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>STOPICP (ALTER and CHANGE MANAGER version 11.2 and later)</td>
<td>ALLOWICP</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>STOPICP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOPCOMM</td>
<td>NOSTOPCOMMIT</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>STOPCOMM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOPLIST</td>
<td>STOPLIST</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>NOSTOPLIST</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STORCLAS</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>SYNCNPNT</td>
<td>SYNCPOINT</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>SYSTYPE</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>TABLEACC</td>
<td>NOTABLEACCESS</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>TABLEACCESS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TABLEALL</td>
<td>NOTABLEALL</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>TABLEALL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC PKPALL</td>
<td>OVERRIDE (KEEP-ALL-PRIMARY)</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td>UNLDCOLL</td>
<td>NOUNLDCOLL</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>UNLDCOLL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Symbolic variables for BMC Administrative products

You can use symbolic variables in the installation options module, the product options file (POF), and on several of the options panels. The symbolic variables are used in job cards and data set names.

Table 18 on page 146 lists all of the symbolic variables that the JCL Generation and Execution components use. Some of these variables are not applicable to each of the BMC Administrative products. The table indicates which products resolve the variables in job cards and data set names. The table also lists the related skeleton library (SLIB) (or ISPF) variable and the corresponding OUTPUT and TEMPLATE descriptor variables.

On product panels, an ampersand (&) must precede the symbolic variable. In the JCL of the installation options module, two ampersands (&&) must precede the symbolic variable. For information about the use of symbolic variables, see the Installation System Reference Manual, Installation System Quick Start, and the BMC Products and Solutions for DB2 Customization Guide.

<table>
<thead>
<tr>
<th>Installation option</th>
<th>Keyword</th>
<th>Type of keyword</th>
<th>Overrides or replaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNLDEMT</td>
<td>NOUNLOADEMPTY</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>UNLOADEMPTY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>USENGTR</td>
<td>USENGTR</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>NOUSENGTR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPDSTATS</td>
<td>ALLSTATSUPD</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>BMCSTATSUPD</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DB2STATSUPD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UTILCOPY</td>
<td>NOUTILCOPY</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>UTILCOPY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VRM</td>
<td>None</td>
<td>None</td>
<td>NA</td>
</tr>
<tr>
<td>VVALPROP</td>
<td>NOVVALPROP</td>
<td>ALUIN</td>
<td>Overrides</td>
</tr>
<tr>
<td></td>
<td>VVALPROP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WDSN</td>
<td>ACM_WDSN</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>WLPS</td>
<td>ACM_WLPS</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>WLSS</td>
<td>ACM_WLSS</td>
<td>POF</td>
<td>Replaces</td>
</tr>
<tr>
<td>WLU</td>
<td>ACM_WLU</td>
<td>POF</td>
<td>Replaces</td>
</tr>
</tbody>
</table>
Table 18: Symbolic variables for BMC Software Administrative products

<table>
<thead>
<tr>
<th>Symbolic variable and description</th>
<th>Size</th>
<th>Value</th>
<th>Related SLIB variable</th>
<th>OUTPUT descriptor variable</th>
<th>TEMPLATE descriptor variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>AJXODS44</td>
<td>1</td>
<td>Y</td>
<td>AJXDSN44</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Whether JCL Generation verifies that the prefix of a data set name contains 44 characters</td>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALID a  b</td>
<td>8</td>
<td>ALTER</td>
<td>AJXJAID</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td>Product ID</td>
<td></td>
<td>CATMGR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DASDMGR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATTACH DB2 subsystem ID</td>
<td>4</td>
<td>None</td>
<td>AJXJSSID</td>
<td>ATTACH</td>
<td>JOBNAME</td>
</tr>
<tr>
<td>BMCCP</td>
<td>2 to 4</td>
<td>None</td>
<td>AJXBMCCP</td>
<td>PART</td>
<td>PART</td>
</tr>
<tr>
<td>Partition number in which insignificant digits are suppressed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>If the partition number is between 0 and 99, the variable will resolve to 2 digits; between 100 and 999, 3 digits; and between 1000 and 4096, 4 digits.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CR</td>
<td>8</td>
<td>None</td>
<td>AJXCR</td>
<td>DB</td>
<td>DB</td>
</tr>
<tr>
<td>Creator</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DA</td>
<td>2</td>
<td>DD</td>
<td>AJXYMD</td>
<td>DAY</td>
<td>DA</td>
</tr>
<tr>
<td>DAY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DAY</td>
</tr>
<tr>
<td>Day part of YYMDDD format</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE a  b</td>
<td>6</td>
<td>YYMDDD</td>
<td>AJXYMD</td>
<td>DATE</td>
<td>DATE</td>
</tr>
<tr>
<td>System date (same format as JYMD and YMD variables)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATEJ</td>
<td>7</td>
<td>YYYYDDD</td>
<td>AJX4YDDD</td>
<td>JDATE</td>
<td>JDATE</td>
</tr>
<tr>
<td>Julian date shown with four-digit year</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DB</td>
<td>8</td>
<td>Name of current database</td>
<td>AJXDB</td>
<td>DB</td>
<td>DB</td>
</tr>
<tr>
<td>DBNAME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Database name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DB2V2</td>
<td>2</td>
<td>Version of DB2</td>
<td>AJXDB2V2</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Symbolic variable and description</td>
<td>Size</td>
<td>Value</td>
<td>Related SLIB variable</td>
<td>OUTPUT descriptor variable</td>
<td>TEMPLATE descriptor variable</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
<td>---------</td>
<td>-----------------------</td>
<td>----------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>DB2V3</td>
<td>3</td>
<td>Version of DB2</td>
<td>AJXDB2V3</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>DDD Julian day</td>
<td>3</td>
<td>DDD</td>
<td>AJXYYDDD</td>
<td>JDAY</td>
<td>JDAY</td>
</tr>
<tr>
<td>DDNAME DDname</td>
<td>8</td>
<td>None</td>
<td>AJXJDDN</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>DDOPT Name of the installation options module</td>
<td>8</td>
<td>None</td>
<td>AJXDDOPT</td>
<td>SEQ</td>
<td>JOBNAME</td>
</tr>
<tr>
<td>DDSEQ Sequence number</td>
<td>4</td>
<td>None</td>
<td>AJXDDSQC</td>
<td>SEQ</td>
<td>SEQ</td>
</tr>
<tr>
<td>DSNUM Partition number</td>
<td>3</td>
<td>None</td>
<td>AJXPARTC</td>
<td>DSNUM</td>
<td>PART</td>
</tr>
<tr>
<td>DT System date (same format as JYMD and YMD)</td>
<td>6</td>
<td>YYMMDD</td>
<td>AJXYMD</td>
<td>DATE</td>
<td>DT</td>
</tr>
<tr>
<td>FCMD Full command name</td>
<td>8</td>
<td>None</td>
<td>AJXFCMD</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td>GDG Generation data group (GDG) counter</td>
<td>4</td>
<td>Initially 1</td>
<td>AJXGDGPC</td>
<td>+1</td>
<td>+1</td>
</tr>
<tr>
<td>GRPNM DB2 subsystem ID</td>
<td>4</td>
<td>None</td>
<td>AJXJSSID</td>
<td>SSID</td>
<td>SSID</td>
</tr>
<tr>
<td>HM Time of JCL creation in hours and minutes</td>
<td>4</td>
<td>HHMM</td>
<td>AJXHM</td>
<td>HOUR.MINUTE</td>
<td>HO.MI</td>
</tr>
<tr>
<td>HMS Time of JCL creation in hours, minutes, and seconds</td>
<td>6</td>
<td>HHMMSS</td>
<td>AJXHMS</td>
<td>TIME</td>
<td>TIME</td>
</tr>
<tr>
<td>HO HOUR Hour part of HHMMSS format</td>
<td>2</td>
<td>HH</td>
<td>AJXHMS</td>
<td>HOUR</td>
<td>HO</td>
</tr>
<tr>
<td>Symbolic variable and description</td>
<td>Size</td>
<td>Value</td>
<td>Related SLIB variable</td>
<td>OUTPUT descriptor variable</td>
<td>TEMPLATE descriptor variable</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
<td>-------</td>
<td>-----------------------</td>
<td>---------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>IC</td>
<td>1</td>
<td>L = Local R = Remote</td>
<td>AJXIC</td>
<td>ICTYPE</td>
<td>IC ICTYPE</td>
</tr>
<tr>
<td>ICTYPE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image copy type</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IS</td>
<td>8</td>
<td>None</td>
<td>AJXIXSPC</td>
<td>TS</td>
<td>TS</td>
</tr>
<tr>
<td>Index space name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IX</td>
<td>16</td>
<td>Name of current index</td>
<td>AJXIX</td>
<td>TS</td>
<td>IS</td>
</tr>
<tr>
<td>IXNAME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IXCR</td>
<td>8</td>
<td>Creator of current index</td>
<td>AJXCR</td>
<td>DB</td>
<td>DB</td>
</tr>
<tr>
<td>Index creator name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IXNODE</td>
<td>22</td>
<td>None</td>
<td>AJXIXNOD</td>
<td>DB.TS</td>
<td>DB..IS</td>
</tr>
<tr>
<td>Index node</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IXSPC</td>
<td>8</td>
<td>None</td>
<td>AJXIXSPC</td>
<td>TS</td>
<td>IS</td>
</tr>
<tr>
<td>Index space name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JD</td>
<td>3</td>
<td>YYDDD</td>
<td>AJXYYDDD</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Julian date shown with two-digit year</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JDATE</td>
<td>7</td>
<td>YYYYDDD</td>
<td>AJX4YDDD</td>
<td>JDATE</td>
<td>JDATE</td>
</tr>
<tr>
<td>Julian date shown with four-digit year</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JDAY</td>
<td>3</td>
<td>DDD</td>
<td>AJXYYDDD</td>
<td>JDAY</td>
<td>JDAY</td>
</tr>
<tr>
<td>Julian day</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JDDN</td>
<td>8</td>
<td>None</td>
<td>AJXDDN</td>
<td>SEQ</td>
<td>JOBNAME</td>
</tr>
<tr>
<td>ddname for skeleton</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JHMS</td>
<td>6</td>
<td>HHMMSS</td>
<td>AJXHMS</td>
<td>TIME</td>
<td>TIME</td>
</tr>
<tr>
<td>Time of work ID creation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JJULD</td>
<td>5</td>
<td>YYDDD</td>
<td>AJXJJULD</td>
<td>JDAY</td>
<td>JDAY</td>
</tr>
<tr>
<td>Julian date of work ID creation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JOBCHAR</td>
<td>1</td>
<td>A = Analysis B = Baseline C = Compare E = Execution I = Import</td>
<td>AJXFJCHR</td>
<td>JOBNAME</td>
<td>JOBNAME</td>
</tr>
<tr>
<td>Component for which JCL is being generated</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Symbolic variable and description</td>
<td>Size</td>
<td>Value</td>
<td>Related SLIB variable</td>
<td>OUTPUT descriptor variable</td>
<td>TEMPLATE descriptor variable</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
<td>--------------------------------------------</td>
<td>-----------------------</td>
<td>-----------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td><strong>JOBNAME</strong></td>
<td>8</td>
<td>Work ID or job name in the JOB statement</td>
<td>AJXWKID AJXJOBNM</td>
<td>JOBNAME</td>
<td>JOBNAME</td>
</tr>
<tr>
<td><strong>JOBTYPE</strong></td>
<td>8</td>
<td>ANALYSIS EXECUTION BASELINE COMPARE IMPORT CATMGR UTILITY DEFINE CAT ALTER CHGMGR DASDMGR</td>
<td>AJXJOBT</td>
<td>TYPE</td>
<td>IC</td>
</tr>
<tr>
<td><strong>JPCOD</strong></td>
<td>3</td>
<td>ACM = CHANGE MANAGER ACT = CATALOG MANAGER ALU = ALTER ASU = DASD MANAGER PLUS</td>
<td>AJXJPCOD</td>
<td>JOBNAME</td>
<td>JOBNAME</td>
</tr>
<tr>
<td><strong>JS1</strong></td>
<td>1</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td><strong>JS2</strong></td>
<td>2</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td><strong>JS4</strong></td>
<td>4</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td><strong>JQID</strong></td>
<td>8</td>
<td>Work ID</td>
<td>AJXJQID</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td><strong>JSSID</strong></td>
<td>4</td>
<td>None</td>
<td>AJXSSID</td>
<td>SSID</td>
<td>SSID</td>
</tr>
<tr>
<td><strong>JU</strong></td>
<td>7</td>
<td>YYYYDDD</td>
<td>AJX4YDDD</td>
<td>JDATE</td>
<td>JDATE</td>
</tr>
<tr>
<td><strong>JUL4Y</strong></td>
<td>7</td>
<td>YYYYDDD</td>
<td>AJX4YDDD</td>
<td>JDATE</td>
<td>JDATE</td>
</tr>
<tr>
<td>Symbolic variable and description</td>
<td>Size</td>
<td>Value</td>
<td>Related SLIB variable</td>
<td>OUTPUT descriptor variable</td>
<td>TEMPLATE descriptor variable</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>------</td>
<td>-------</td>
<td>-----------------------</td>
<td>---------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>JULIAN</td>
<td>5</td>
<td>YYDDD</td>
<td>AJXYYDDD</td>
<td>JDATE</td>
<td>JDATE</td>
</tr>
<tr>
<td>System date</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JYMD</td>
<td>6</td>
<td>YYMMDD</td>
<td>AJXJYMD</td>
<td>DATE</td>
<td>DATE</td>
</tr>
<tr>
<td>Date of work ID creation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(same as DATE)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LDSNUM</td>
<td>3</td>
<td>None</td>
<td>AJXPARTC</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Partition number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LI</td>
<td>8</td>
<td>None</td>
<td>AJXLDEFN</td>
<td>UTIL</td>
<td>LI</td>
</tr>
<tr>
<td>LIST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LIST</td>
</tr>
<tr>
<td>LISTDEF name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LLQ</td>
<td>4</td>
<td>None</td>
<td>AJXLLQ</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Low-level qualifier for ISPF data sets</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOCREM</td>
<td>1</td>
<td>L = Local</td>
<td>AJXLR</td>
<td>TYPE</td>
<td>LOCREM</td>
</tr>
<tr>
<td>Image copy type</td>
<td></td>
<td>R = Remote</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LPART</td>
<td>3</td>
<td>None</td>
<td>AJXPARTC</td>
<td>LPART</td>
<td>PART</td>
</tr>
<tr>
<td>Partition number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LR</td>
<td>1</td>
<td>L = Local</td>
<td>AJXLR</td>
<td>TYPE</td>
<td>LR</td>
</tr>
<tr>
<td>Image copy type</td>
<td></td>
<td>R = Remote</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MEMBER</td>
<td>8</td>
<td>None</td>
<td>AJXMEMBER</td>
<td>JOBNAME</td>
<td>JOBNAME</td>
</tr>
<tr>
<td>MEMBR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PDS member name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Note: If the work ID template is used in CM/PILOT, then the task ID is used.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Symbolic variable and description</td>
<td>Size</td>
<td>Value</td>
<td>Related SLIB variable</td>
<td>OUTPUT descriptor variable</td>
<td>TEMPLATE descriptor variable</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
<td>-------</td>
<td>-----------------------</td>
<td>-----------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>MI MINUTE</td>
<td>2</td>
<td>MM</td>
<td>AJXHMS</td>
<td>MINUTE</td>
<td>MI MINUTE</td>
</tr>
<tr>
<td>Minute part of HHMMSS format</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MMDD</td>
<td>4</td>
<td>MMDD</td>
<td></td>
<td>MONTH.DAY</td>
<td>MONTH.DAY</td>
</tr>
<tr>
<td>JCL date</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MO MONTH</td>
<td>2</td>
<td>MM</td>
<td>AJXYMD</td>
<td>MONTH</td>
<td>MO MONTH</td>
</tr>
<tr>
<td>Month part of YYMMDD format</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSSID DB2 subsystem ID</td>
<td>4</td>
<td></td>
<td>AJXMSSID</td>
<td>SSID</td>
<td>SSID</td>
</tr>
<tr>
<td>Determined at runtime from the</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-JCLP command in the worklist</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OBJT OBJTYP Object type</td>
<td>2</td>
<td>Object type: TS, TB, IX</td>
<td>AJXOBT</td>
<td>UTIL TYPE</td>
<td>IC</td>
</tr>
<tr>
<td>OBNAM Object name</td>
<td>27</td>
<td>None</td>
<td>AJXOBNAM</td>
<td>DB.TS CREATOR.IX</td>
<td>DB..SN</td>
</tr>
<tr>
<td>OBNOD Database and either table</td>
<td>17</td>
<td>#### (for objects</td>
<td>AJXOBNOD</td>
<td>DB.TS</td>
<td>DB..SN</td>
</tr>
<tr>
<td>space name or index space name</td>
<td></td>
<td>that do not exist)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PA Partition number</td>
<td>3</td>
<td>None</td>
<td>AJXPARTC</td>
<td>PART</td>
<td>PA</td>
</tr>
<tr>
<td>PART Partition number in which</td>
<td>4</td>
<td>None</td>
<td>AJXPARTC</td>
<td>PART</td>
<td>PART</td>
</tr>
<tr>
<td>insignificant digits are</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>suppressed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PART4 Partition number in which</td>
<td>4</td>
<td>None</td>
<td>AJX4PART</td>
<td>PART</td>
<td>PART</td>
</tr>
<tr>
<td>leading zeros are not suppressed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PART5 Partition number in which</td>
<td>5</td>
<td>None</td>
<td>AJX5PART</td>
<td>PART</td>
<td>PART</td>
</tr>
<tr>
<td>leading zeros are not suppressed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Symbolic variable and description</td>
<td>Size</td>
<td>Value</td>
<td>Related SLIB variable</td>
<td>OUTPUT descriptor variable</td>
<td>TEMPLATE descriptor variable</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------</td>
<td>-------</td>
<td>------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>PB PRIBAC Type of copy</td>
<td>1</td>
<td>P = Primary B = Backup</td>
<td>AJXPB</td>
<td>TYPE</td>
<td>PB PRIBAC</td>
</tr>
<tr>
<td>PGMR    Programmer name field</td>
<td>20</td>
<td>'&amp;&amp;JOBTYP - &amp;&amp;WKID'</td>
<td>AJXPGMR</td>
<td>JOBNAME</td>
<td>JOBNAME</td>
</tr>
<tr>
<td>PREFIX  (ALTER and CHANGE MANAGER) TSO prefix (same as ZPREFIX)</td>
<td>8</td>
<td>TSO prefix (&amp;ZUSER if NOPREFIX) or user ID</td>
<td>ZPREFIX</td>
<td>None</td>
<td><strong>PREFIX</strong></td>
</tr>
<tr>
<td>RHLQ    High-level qualifier</td>
<td>60</td>
<td>High-level qualifier (HLQ) for ISPF data sets</td>
<td>AJXRHLQ</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>RSEQ#   Restart sequence number</td>
<td>6</td>
<td>None</td>
<td>AJXRSEQ#</td>
<td>SEQ</td>
<td>SEQ</td>
</tr>
<tr>
<td>RTYPE   Object type</td>
<td>2</td>
<td>None</td>
<td>AJXOBJT</td>
<td>TYPE</td>
<td>IC</td>
</tr>
<tr>
<td>RUNTP   Type of run</td>
<td>9</td>
<td>ANALYZE EXECUTE BASELINE COMPARE IMPORT ALTER RESTART STARTOVER</td>
<td>AJXRUNTP</td>
<td>TYPE</td>
<td>IC</td>
</tr>
<tr>
<td>SC      SEC SECOND Second part of HHMMSS format</td>
<td>2</td>
<td>SS</td>
<td>AJXHMS</td>
<td>SEC</td>
<td>SC</td>
</tr>
<tr>
<td>SEQ     DD sequence number</td>
<td>4</td>
<td>None</td>
<td>AJXDDSQC</td>
<td>SEQ</td>
<td>SEQ</td>
</tr>
<tr>
<td>SEQ#    Sequence number</td>
<td>6</td>
<td>None</td>
<td>AJXSEQ#</td>
<td>SEQ</td>
<td>SEQ</td>
</tr>
<tr>
<td>SN      Table space name or index space name</td>
<td>8</td>
<td>None</td>
<td>AJXSPNAM</td>
<td>TS</td>
<td>SN</td>
</tr>
<tr>
<td>Symbolic variable and description</td>
<td>Size</td>
<td>Value</td>
<td>Related SLIB variable</td>
<td>OUTPUT descriptor variable</td>
<td>TEMPLATE descriptor variable</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
<td>---------</td>
<td>-----------------------</td>
<td>-----------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>SPNAME</td>
<td>8</td>
<td>None</td>
<td>AJXSPNAM</td>
<td>TS</td>
<td>SN</td>
</tr>
<tr>
<td>Table space name or index space name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQ</td>
<td>4</td>
<td>None</td>
<td>AJXDDSQC</td>
<td>SEQ</td>
<td>SQ</td>
</tr>
<tr>
<td>DD sequence number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SS</td>
<td>4</td>
<td>None</td>
<td>AJXJSSID</td>
<td>SSID</td>
<td>SS</td>
</tr>
<tr>
<td>DB2 subsystem ID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSID</td>
<td>4</td>
<td>Determined at runtime</td>
<td>AJXSSID</td>
<td>SSID</td>
<td>SSID</td>
</tr>
<tr>
<td>DB2 subsystem ID (same as JSSID)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>8</td>
<td>None</td>
<td>AJXSTEPN</td>
<td>STEPNAME</td>
<td>STEPNAME</td>
</tr>
<tr>
<td>STEPN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STEPNAME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST#</td>
<td>6</td>
<td>None</td>
<td>AJXSTEPC</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Step number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSMLIB</td>
<td>46</td>
<td>Name of system ISPF message library</td>
<td>AJXSYSM</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>System ISPF message library</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSUID</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td><strong>PREFIX</strong></td>
</tr>
<tr>
<td>JCL symbolic parameter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TBCR</td>
<td>8</td>
<td>Owner of current table</td>
<td>AJXTBCR</td>
<td>DB</td>
<td>DB</td>
</tr>
<tr>
<td>TBCRE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Table creator name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TBNAM</td>
<td>12</td>
<td>Name of current table</td>
<td>AJXTBNAM</td>
<td>TS</td>
<td>TS</td>
</tr>
<tr>
<td>d TBNAME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Table name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TBNODE</td>
<td>22</td>
<td>None</td>
<td>AJXTBNOD</td>
<td>DB.TS</td>
<td>DB..TS</td>
</tr>
<tr>
<td>Table node</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TI</td>
<td>6</td>
<td>HHMMSS</td>
<td>AJXHMS</td>
<td>TIME</td>
<td>TI TIME</td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME4</td>
<td>4</td>
<td>HHMM</td>
<td>AJXHMS</td>
<td>HOUR.MINUTE</td>
<td>HO.MI</td>
</tr>
<tr>
<td>Hours and minutes of HHMMSS format</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Symbolic variable and description</td>
<td>Size</td>
<td>Value</td>
<td>Related SLIB variable</td>
<td>OUTPUT descriptor variable</td>
<td>TEMPLATE descriptor variable</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
<td>------------------------------------------</td>
<td>-----------------------</td>
<td>---------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>TS</td>
<td>8</td>
<td>Name of current table space</td>
<td>AjxTS</td>
<td>TS</td>
<td>TS</td>
</tr>
<tr>
<td>TSNANE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Table space name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSCR</td>
<td>8</td>
<td>None</td>
<td>AjxTSCR</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Table space creator</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSIX</td>
<td>8</td>
<td>None</td>
<td>AjxSPNAM</td>
<td>TS</td>
<td>SN</td>
</tr>
<tr>
<td>Table space name or index space name</td>
<td>8</td>
<td>None</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSSID</td>
<td>4</td>
<td>Determined at runtime from the -JCLP command in the worklist</td>
<td>AjxtSSID</td>
<td>SSID</td>
<td>SSID</td>
</tr>
<tr>
<td>DB2 subsystem ID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TU1</td>
<td>4</td>
<td>None</td>
<td>AjxTU1</td>
<td>STEPNAME</td>
<td>STEPNAME</td>
</tr>
<tr>
<td>Tape unit 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TU2</td>
<td>4</td>
<td>None</td>
<td>AjxTU2</td>
<td>STEPNAME</td>
<td>STEPNAME</td>
</tr>
<tr>
<td>Tape unit 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TU3</td>
<td>4</td>
<td>None</td>
<td>AjxTU3</td>
<td>STEPNAME</td>
<td>STEPNAME</td>
</tr>
<tr>
<td>Tape unit 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TYPE</td>
<td>1</td>
<td>L = Local R = Remote</td>
<td>AjxLR</td>
<td>TYPE</td>
<td>IC</td>
</tr>
<tr>
<td>Type of copy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCMD</td>
<td>4</td>
<td>None</td>
<td>AjxUCMD</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td>Command ID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UDOPT</td>
<td>8</td>
<td>None</td>
<td>AjxUDOPT</td>
<td>STEPNAME</td>
<td>STEPNAME</td>
</tr>
<tr>
<td>Installation options file lName</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UID</td>
<td>7</td>
<td>None</td>
<td>ZUSER</td>
<td>None</td>
<td><strong>PREFIX</strong></td>
</tr>
<tr>
<td>TSO user ID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ULLQ</td>
<td>4</td>
<td>Low-level qualifier for user-defined data sets</td>
<td>AjxULLQ</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Low-level qualifier for user-defined data sets</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Symbolic variable and description</td>
<td>Size</td>
<td>Value</td>
<td>Related SLIB variable</td>
<td>OUTPUT descriptor variable</td>
<td>TEMPLATE descriptor variable</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------</td>
<td>-------</td>
<td>-----------------------</td>
<td>-----------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>UP UPART</td>
<td>3</td>
<td>None</td>
<td>AJXUPART</td>
<td>PART</td>
<td>PART</td>
</tr>
<tr>
<td>Partition number variable compared to AJXPARTC in which insignificant digits are suppressed For example, if a partition number is 10, AJXUPART will contain 010, while AJXPARTC will contain 10.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USER1 User-defined</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td>USER2 User-defined</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td>USERID a b TSO user ID</td>
<td>7</td>
<td>TSO user ID</td>
<td>ZUSER</td>
<td>USERID</td>
<td><strong>PREFIX</strong></td>
</tr>
<tr>
<td>UT UTID UTIL UTILID Utility ID</td>
<td>16</td>
<td>None</td>
<td>AJXUTID</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td>UILPFX First eight bytes of utility ID</td>
<td>8</td>
<td>None</td>
<td>AJXUTID</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td>UTILSFX Last eight bytes of utility ID</td>
<td>8</td>
<td>None</td>
<td>AJXUTID</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td>UVR1 UVR2 UVR3 UVR4 UVR5 User-defined character variable</td>
<td>8</td>
<td>User-defined variable or # # # # # # (if value is blank)</td>
<td>AJXUVR1 AJXUVR2 AJXUVR3 AJXUVR4 AJXUVR5</td>
<td>User-defined</td>
<td>User-defined</td>
</tr>
<tr>
<td>VCAT VCAT name</td>
<td>8</td>
<td>None</td>
<td>AJXVCAT</td>
<td>DB</td>
<td>DB</td>
</tr>
<tr>
<td>Symbolic variable and description</td>
<td>Size</td>
<td>Value</td>
<td>Related SLIB variable</td>
<td>OUTPUT descriptor variable</td>
<td>TEMPLATE descriptor variable</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
<td>-------</td>
<td>------------------------</td>
<td>----------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>(DASD MANAGER PLUS)WKID</td>
<td>8</td>
<td>Name of the current work ID in use</td>
<td>AJXJQID</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td>WKOWN a</td>
<td>8</td>
<td>Owner of the current work ID in use</td>
<td>AJXWKOWN</td>
<td>JOBNAME STEPNAME</td>
<td>JOBNAME STEPNAME</td>
</tr>
<tr>
<td>WKOWNER a</td>
<td></td>
<td>Work ID owner</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WORKID a</td>
<td>18</td>
<td>Name of the current work ID in use or, for Compare only, the outbound migrate profile name specified that is for use. If the work ID name contains characters that are invalid for use in data set names, the work ID will be truncated at the first invalid character.</td>
<td>AJXJQID</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td>WORKID8 a b</td>
<td>8</td>
<td>First eight characters of the work ID name</td>
<td>AJXWKID</td>
<td>UTIL</td>
<td>UT</td>
</tr>
<tr>
<td>YE</td>
<td>4</td>
<td>YYYY</td>
<td>AJX4YDDD</td>
<td>YEAR</td>
<td>YE</td>
</tr>
<tr>
<td>YEAR a</td>
<td></td>
<td>Year from Julian date</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YMD</td>
<td>6</td>
<td>YYMMDD</td>
<td>AJXYMD</td>
<td>DATE</td>
<td>DATE</td>
</tr>
<tr>
<td>YMD a</td>
<td></td>
<td>Date of JCL creation (same as DATE and JYMD)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YY</td>
<td>2</td>
<td>YY</td>
<td>AJXYYDDD</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>YYDDD a</td>
<td>5</td>
<td>YYDDD</td>
<td>AJXYYDDD</td>
<td>JDATE</td>
<td>JDATE</td>
</tr>
<tr>
<td>YYDDD b</td>
<td></td>
<td>Julian date of JCL creation (same as JULIAN)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Symbolic variables for BMC Administrative products

ALTER and CHANGE MANAGER for DB2 Reference Manual
<table>
<thead>
<tr>
<th>Symbolic variable and description</th>
<th>Size</th>
<th>Value</th>
<th>Related SLIB variable</th>
<th>OUTPUT descriptor variable</th>
<th>TEMPLATE descriptor variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>YYYYDDD</td>
<td>7</td>
<td>YYYYDDD</td>
<td>AJX4YDDD</td>
<td>JDATE</td>
<td>JDATE</td>
</tr>
<tr>
<td>Julian date shown with four-digit year</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZACCTNUM</td>
<td>40</td>
<td>If the replacement value is not known or does not fit in the space provided, question marks (?) are substituted.</td>
<td>ZACCTNUM</td>
<td>USERID</td>
<td>JOBNAME</td>
</tr>
<tr>
<td>User’s account number for jobs that are generated by the product</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZPREFIX a b g</td>
<td>8</td>
<td>None</td>
<td>ZPREFIX</td>
<td>USERID</td>
<td><strong>PREFIX</strong></td>
</tr>
<tr>
<td>TSO prefix</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZSYSID</td>
<td></td>
<td>System name</td>
<td>ZSYSID</td>
<td>ATTACH</td>
<td><strong>PREFIX</strong></td>
</tr>
<tr>
<td>ISPF system variable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZUSER a b</td>
<td>8</td>
<td>None</td>
<td>ZUSER</td>
<td>USERID</td>
<td><strong>PREFIX</strong></td>
</tr>
<tr>
<td>User ID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

a ALTER and CHANGE MANAGER resolve this variable for job cards and data set names.

b DASD MANAGER PLUS resolves this variable for job cards and data set names.

c This variable is incremented each time any image copy is taken for a specific table space. Consider the following items if you use the GDG variable:

- JCL Generation does not verify that the GDG base definitions already exist.
- No GDG numbers are built for invalid or incomplete table space names. JCL Generation builds the &AJXDB, &AJXTS, and &OBNOD variables by parsing the utility commands in the worklist. An alter-type worklist might contain incomplete table space names for implicit table spaces because these names will not be known until the worklist is executed.

d This value truncates after eight characters when used by JCL Generation.

e This value is determined at runtime (same as SSID).

f For the Compare component of CHANGE MANAGER, this variable indicates the outbound migrate profile name.

g Leave this variable blank for NOPREFIX (same as PREFIX).
ALTER and CHANGE MANAGER installation options

The installation process creates the installation options module for the ALTER and CHANGE MANAGER products. This module resides in $xnnDOPT and also in the HLQ:UBMCCNTL member (where HLQ is the high-level qualifier) that has the same name as the installation options module. The module's default name is ACTDOPD1.

Note

The installation options are also called default options, or DOPTs.

This section provides an example of the module and descriptions of each option.

Installation options

The ALTER and CHANGE MANAGER installation options are also known as default options, or DOPTs.

Figure 1 on page 159 provides an example of the installation options module for CHANGE MANAGER.

Figure 1: CHANGE MANAGER installation options module

File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help

---MSG> -Warning- The UNDO command is not available until you change your edit profile using the command RECOVERY ON.

---MSG> - Top of Data -------------------------------

000001 *DOPTS NAME : DMC1QEJM
000002 *DATE DOPTS ASSEMBLED: 20160616
000003 ACMDOPTS CSECT ,
000004 ACMDOPTS RMODE 24
000005 ACMDOPTS AMODE 24
000006 ACMDOPTS $ALUOPT PRODUCT='CHANGE MANAGER',
000007 DATE=&SYSDATC,
000008 PC=ACM,
000009 EURO=(N,R),
000010 SYSTYPE=S,
000011 LOG=N,
000012 DBRMLIB=N,
<table>
<thead>
<tr>
<th>Line</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>000013</td>
<td>WDSN=</td>
</tr>
<tr>
<td>000014</td>
<td>CATAUDIT=(N,R)</td>
</tr>
<tr>
<td>000015</td>
<td>CATRECOV=(N,R)</td>
</tr>
<tr>
<td>000016</td>
<td>SEQI=050,</td>
</tr>
<tr>
<td>000017</td>
<td>SYNCPT=10,</td>
</tr>
<tr>
<td>000018</td>
<td>ALLOC=N,</td>
</tr>
<tr>
<td>000019</td>
<td>STORCLAS=N,</td>
</tr>
<tr>
<td>000020</td>
<td>DATACLAS=N,</td>
</tr>
<tr>
<td>000021</td>
<td>MGMTCLAS=N,</td>
</tr>
<tr>
<td>000022</td>
<td>AUTHSW=(N,R),</td>
</tr>
<tr>
<td>000023</td>
<td>DASMDM=(Y,R),</td>
</tr>
<tr>
<td>000024</td>
<td>CCSID=(E,R),</td>
</tr>
<tr>
<td>000025</td>
<td>VVALPROP=(N,R),</td>
</tr>
<tr>
<td>000026</td>
<td>BPOOLTS=BPO,</td>
</tr>
<tr>
<td>000027</td>
<td>BPOOILX=BPO,</td>
</tr>
<tr>
<td>000028</td>
<td>DISCARDS=(0001,R),</td>
</tr>
<tr>
<td>000029</td>
<td>BMCCOPY=(N,R),</td>
</tr>
<tr>
<td>000030</td>
<td>BMCCHECK=(N,R),</td>
</tr>
<tr>
<td>000031</td>
<td>BMLOAD=(N,R),</td>
</tr>
<tr>
<td>000032</td>
<td>UTILCOPY=(N,R),</td>
</tr>
<tr>
<td>000033</td>
<td>BMCLUD=(I,R),</td>
</tr>
<tr>
<td>000034</td>
<td>REORG=(I,R),</td>
</tr>
<tr>
<td>000035</td>
<td>REBLD=(I,R),</td>
</tr>
<tr>
<td>000036</td>
<td>UNLDCOLL=N,</td>
</tr>
<tr>
<td>000037</td>
<td>PARTCPY=N,</td>
</tr>
<tr>
<td>000038</td>
<td>PARALLEL=N,</td>
</tr>
<tr>
<td>000039</td>
<td>ZPARM=DSNZPARM,</td>
</tr>
<tr>
<td>000040</td>
<td>MAXSYSUT=20,</td>
</tr>
<tr>
<td>000041</td>
<td>BMCFASTL=Y,</td>
</tr>
<tr>
<td>000042</td>
<td>DYNCOPY=Y,</td>
</tr>
<tr>
<td>000043</td>
<td>DYNCLUD=Y,</td>
</tr>
<tr>
<td>000044</td>
<td>STATS=(S,R),</td>
</tr>
<tr>
<td>000045</td>
<td>UPDSTATS=(C,R),</td>
</tr>
<tr>
<td>000046</td>
<td>TABLEALL=(Y,R),</td>
</tr>
<tr>
<td>000047</td>
<td>UNLDEMT=(Y,R),</td>
</tr>
<tr>
<td>000048</td>
<td>STOPCOMM=(N,R),</td>
</tr>
<tr>
<td>000049</td>
<td>TABLEACC=(Y,R),</td>
</tr>
<tr>
<td>000050</td>
<td>KEYCARD=N,</td>
</tr>
<tr>
<td>000051</td>
<td>GENOBD=N,</td>
</tr>
<tr>
<td>000052</td>
<td>COPYYD01=R,</td>
</tr>
<tr>
<td>000053</td>
<td>COPYYD02=N,</td>
</tr>
<tr>
<td>000054</td>
<td>HSMVOL=,</td>
</tr>
<tr>
<td>000055</td>
<td>ENV=ACMC1QDE,</td>
</tr>
<tr>
<td>000056</td>
<td>FEP=ACMC1QDF,</td>
</tr>
<tr>
<td>000057</td>
<td>SPP=ACMC1QDS,</td>
</tr>
<tr>
<td>000058</td>
<td>ANP=ACMC1QDA,</td>
</tr>
<tr>
<td>000059</td>
<td>IMP=ACMC1QDI,</td>
</tr>
<tr>
<td>000060</td>
<td>CMP=ACMC1QDC,</td>
</tr>
<tr>
<td>000061</td>
<td>BASE=ACMC1QDB,</td>
</tr>
<tr>
<td>000062</td>
<td>RPTPL=ACMC1QDR,</td>
</tr>
<tr>
<td>000063</td>
<td>EPP=ACMC1QXM,</td>
</tr>
<tr>
<td>000064</td>
<td>EAP=ACMC1QXA,</td>
</tr>
<tr>
<td>000065</td>
<td>EIP=TISINSTL,</td>
</tr>
<tr>
<td>000066</td>
<td>DEFERUIX='N',</td>
</tr>
<tr>
<td>000067</td>
<td>REORGALT=(N,R),</td>
</tr>
<tr>
<td>000068</td>
<td>POFDS=('AUS.DOPSEC.CNTL(JXCC1QFJEM)',R),</td>
</tr>
<tr>
<td>000069</td>
<td>STATHIST=N,</td>
</tr>
<tr>
<td>000070</td>
<td>TBLANKS=2,</td>
</tr>
<tr>
<td>000071</td>
<td>REORGALL=(Y,R),</td>
</tr>
<tr>
<td>000072</td>
<td>REORGREF=(Y,R),</td>
</tr>
<tr>
<td>000073</td>
<td>REGENNSP=N,</td>
</tr>
<tr>
<td>000074</td>
<td>DYNREORG=Y,</td>
</tr>
<tr>
<td>000075</td>
<td>CLONDATA=Y,</td>
</tr>
<tr>
<td>000076</td>
<td>BLRTERM='#',</td>
</tr>
<tr>
<td>000077</td>
<td>UCPKPALL=N,</td>
</tr>
<tr>
<td>000078</td>
<td>CMSPTXT=N,</td>
</tr>
<tr>
<td>000079</td>
<td>STOPLIST=.</td>
</tr>
</tbody>
</table>
Note

R in the variable syntax indicates that the value specified will refresh the existing value of the variable in the user’s ISPF profile data set, if the time stamp of the installation options module is later than the time stamp in the user’s ISPF profile member.

Descriptions of the options that are listed in Figure 1 on page 159 follow. In some cases, the default value for the option is listed.

Description of installation options

This section describes each option in the installation options module. In some cases, the default value for the option is listed with the option name.

ALLOC=N

This option indicates the allocation units to use for data sets that are managed by the IBM Storage Management Subsystem (SMS). If the AMS is set to Y, this option determines how the product allocates space for VCAT-defined DB2 objects that SMS manages.

The parameters are defined as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Cylinders</td>
</tr>
<tr>
<td>K</td>
<td>Kilobytes</td>
</tr>
<tr>
<td>M</td>
<td>Megabytes</td>
</tr>
<tr>
<td>N</td>
<td>SMS not in use (default)</td>
</tr>
<tr>
<td>T</td>
<td>Tracks</td>
</tr>
</tbody>
</table>
ANP=ACMervyA

This option defines the Analysis plan name.

AUTHSW=N

This option controls the method of authorization-ID switching that Analysis uses:

- If AUTHSW=N, the product generates the -SETS command in the worklist to change the authorization IDs with SET CURRENT SQLID statements. For additional security, you cannot edit -AUTH commands into the worklist. This option uses DB2 secondary authorization IDs.

**Note**

BMC recommends that you specify **AUTHSW=N**.

- If AUTHSW=Y, the product generates the -AUTH command in the worklist to change the authorization IDs for subsequent SQL statements and reBIND commands. You can also manually add -SETS commands to the worklist for setting the authorization IDs.

The -AUTH command that is generated in the worklist cannot switch to 8-byte authorization IDs. Because the -AUTH command simulates the user ID environment inherent to MVS, and MVS itself is limited to seven characters for user IDs, the authorization ID must be seven characters or less.

If your DB2 environment has committed to the use of SET CURRENT SQLID, the best way to ensure that 8-byte IDs are correctly used is to set the installation option for AUTHSW to N. When AUTHSW=N, the product generates -SETS commands in the worklist instead of -AUTH commands. Using -SETS commands exploits DB2's built-in switching mechanism and the commands are not limited to 7-byte IDs.

If using -SETS commands is not suitable for your environment, you can switch -AUTH commands to 8-byte IDs through the Execution security exit. This exit has many possible uses, including the ability to examine and optionally change the authorization ID that the -AUTH command uses. Through this exit you can cause an 8-byte ID to switch to a corresponding 7-byte (or shorter) ID, which causes a switch in the DB2 DSN3@TH exit that is already installed. In the HLQ.DBSAMP data set, member ALUEUSX1 provides an example of the Execution security exit.

For more information about the Execution security exit, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

**Note**

Use AUTHSW=Y only if your site does not use DB2 secondary authorization IDs.
If AUTHSW=B, the product generates both -AUTH and -SETS commands in the worklist to change the authorization IDs. The product generates -AUTH commands to set the original CREATEDBY values and to DROP objects. The CREATEDBY field in the DB2 catalog remains unchanged after updates. The product generates -SETS commands to set new OWNER values for all objects. In addition, when AUTHSW=B, the product changes the authorization IDs before CREATE TABLE and CREATE INDEX statements.

**WARNING**

Setting AUTHSW=B is not recommended because of a potential security exposure and the additional complexity if you try to restart the worklist. The security exposure exists because the DB2 catalog does not accurately reflect the primary authorization ID of the creator of the objects. If you must set AUTHSW=B, use the sample security exit (ALUEUSX1 in the HLQ.DBSAMP data set) to avoid the security exposure.

If AUTHSW=X, the product disables authorization switching for a worklist. The product does not generate -AUTH, -SETS, and -GLID commands in the worklist.

**Note**

Use AUTHSW=X only under the following conditions:

— You are not allowed to execute any -AUTH or -SETS commands in your environment

— You previously edited your worklists and removed all of the -AUTH, -SETS, and -GLID commands

If AUTHSW=G, the products disable authorization switching for a worklist. The products do not generate -AUTH and -SETS commands; they do, however, generate -GLID commands in the worklist.

For a list of the authorizations that are used for authorization switching, as well as information about -AUTH and -SETS commands, see the *ALTER and CHANGE MANAGER for DB2 Reference Manual*.

**BASE=ACM\_vryB**

*(CHANGE MANAGER only)* This option defines the name of the Baseline plan.

**BINDONIX=N**

After you add a new or alter an index, this option indicates whether to automatically rebind all packages that are dependent on the table containing the index.
BLRTERMC=#

(CHANGE MANAGER only) This option specifies the character that the baseline report uses to separate SQL statements. The default value is a pound sign (#). The value can be any character except a blank, comma (,), double quotation mark ("), single quotation mark (‘), left parenthesis ([], right parenthesis []), or underscore (_).

BMCCHECK=N

This option indicates whether to use the CHECK PLUS utility in place of the IBM CHECK DATA utility for checking referential constraint violations in DB2 table spaces (Y or N).

BMCCOPY=N

This option indicates whether to use the NGT Copy utility in place of the IBM COPY utility. The parameters are defined as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Uses the IBM COPY utility to copy objects. When BMCCOPY=N, the COPYDD01, COPYDD02, RECVDD01, and RECVDD02 installation options must equal R.</td>
</tr>
<tr>
<td>Y</td>
<td>Uses the NGT Copy utility. The NGT Copy utility must be installed. When CHANGE MANAGER is used as a component of the Database Administration for DB2, BMC Next Generation Technology Database Administration for DB2, or BMC Object Administration for DB2 solution, the default is Y.</td>
</tr>
<tr>
<td>X</td>
<td>Does not include a copy utility command in a worklist.</td>
</tr>
<tr>
<td>F</td>
<td>Does not include a copy utility command in a worklist and start objects in copy pending status with ACCESS (FORCE).</td>
</tr>
</tbody>
</table>

BMCFASTL=N

This option indicates whether to use the FORMAT BMCLOAD option (UNLOAD PLUS utility) and FORMAT BMCUNLOAD (LOADPLUS utility) to unload data from one table and load it into another table that has a similar structure (Y or N). If you use CHANGE MANAGER as a component of the Database Administration for DB2, or BMC Object Administration for DB2 solution, the default is Y.

BMCLOAD=N

This option indicates whether to use the LOADPLUS utility for loads instead of the IBM LOAD utility (Y or N). If you use CHANGE MANAGER as a component of the Database Administration for DB2, or BMC Object Administration for DB2 solution, the default is Y.
BMCUNLD=N

This option indicates whether to use the UNLOAD PLUS utility to unload data. The parameters are defined as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Uses the BASIC UNLOAD utility to unload data</td>
</tr>
<tr>
<td>Y</td>
<td>Uses the UNLOAD PLUS utility to unload data. When CHANGE MANAGER is used as a component of the Database Administration for DB2, or BMC Object Administration for DB2 solution, the default is Y.</td>
</tr>
<tr>
<td>I</td>
<td>Uses the IBM UNLOAD utility to unload data</td>
</tr>
</tbody>
</table>

BPOOLIX=BP0

This option indicates the buffer pool for user indexes. Valid values include:
- BP0 through BP49
- BP8K0 through BP8K9
- BP16K0 through BP16K9
- BP32K, BP32K1 through BP32K9

**Note**
In CHANGE MANAGER, the Compare component uses the value of BPOOLIX on the local DB2 subsystem when a remote DB2 catalog or a baseline is used in a comparison.

BPOOLTS=BP0

This option indicates the buffer pool for user data. Valid values include:
- BP0 through BP49
- BP8K0 through BP8K9
- BP16K0 through BP16K9
- BP32K, BP32K1 through BP32K9

The value should match the value specified for the DB2 initialization parameter module, DSNZPARM, on the DB2 subsystem on which the option is used.

**Note**
In CHANGE MANAGER, the Compare component uses the value of BPOOLTS on the local DB2 subsystem when a remote DB2 catalog or a baseline is used in a comparison.
CATAUDIT=N

This option specifies the data definition language (DDL) audit logging indicator. If you have CHANGE MANAGER installed, an entry of Y causes Execution to log executed DDL statements in the CHANGE MANAGER DDL Audit Log (Y or N). For information about the auditing log, see the CATALOG MANAGER for DB2 User Guide.

CATRECOV=N

This option specifies the Drop Recovery indicator. This option is useful only if you have CHANGE MANAGER installed. Type Y if you want the Execution component to invoke CHANGE MANAGER to log recovery information in the CHANGE MANAGER drop-recovery tables for the objects that are dropped when the work ID is executed. For information about drop recovery, see the CATALOG MANAGER for DB2 User Guide.

CCSID=E

This option provides the default encoding scheme for databases that are created using the product.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ASCII</td>
</tr>
<tr>
<td>E</td>
<td>EBCDIC</td>
</tr>
<tr>
<td>U</td>
<td>Unicode</td>
</tr>
</tbody>
</table>

CLONDATA=Y

This option indicates whether to include worklist commands to unload, load, and copy data contained in clone tables (Y or N).

CMP=ACM\textit{very}C

\textit{(CHANGE MANAGER only)} This option defines the Compare plan name.

CMPLIMIT1

Reserved for future use.

CMPLIMIT2

Reserved for future use.

CMPSCWBF

Reserved for future use.
Reserved for future use.

CMPSPTXT=N

This option indicates how the Compare component should compare the text of a native SQL stored procedure.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Ignores extra blanks, blank lines, and character case (uppercase and lowercase) when comparing the text.</td>
</tr>
<tr>
<td>Y</td>
<td>Compares each character in the text verbatim.</td>
</tr>
</tbody>
</table>

COPYDD01=R  
COPYDD02=N

These options define image copies for the BMC and IBM copy utilities. The parameters are defined as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Does not create an image copy.</td>
</tr>
<tr>
<td>C</td>
<td>Creates an image copy of the primary data set at the local site, but does not register the image copy. The NGT Copy, REORG PLUS, or LOADPLUS utility is used to create the copy.</td>
</tr>
</tbody>
</table>
| R         | Creates an image copy of the primary data set at the local site using one of the following utilities:  
  - NGT Copy  
  - REORG PLUS  
  - LOADPLUS  
  - IBM COPY  
  The utility will register the copy.  
  If the IBM COPY utility is used, COPYDD01 and COPYDD02 must equal R. |

DASDMAN=Y

This option indicates whether DASD MANAGER PLUS is installed (Y or N). The product selects DB2 catalog statistics for space estimation. When DASDMAN=Y, any statistics from the BMCSTATS tables are merged.

DATACLAS=N

This option indicates whether support for the DATACLAS parameter is required for VCAT-defined DB2 objects (Y or N).
DATE=&SYSDATC

This option indicates a parameter that is used only if you have ASMA90 as your assembler.

DBRMLIB=N

This option includes the LIBRARY parameter on the BIND statement for plans and packages (Y or N).

---

**Note**

A disadvantage to adding the LIBRARY parameter to the BIND PLAN command is that the order of the libraries on the BIND could be incorrect. If some DBRMs are present in multiple libraries, the product cannot guarantee that the concatenation will result in every DBRM coming from the correct library.

---

DEFERUIX=N

This option enables the product to create unique indexes with the DEFER YES parameter (Y or N).

DISCARDS=nnnn

This option is used by the product to specify the number of discard records to allow. The parameter nnnn specifies the number of discards in a range from 0 to 9999. DISCARDS=0 means that no maximum number of discards exists.

With DISCARDS=1, the product generates one discard DD, //SYSDS001, for the entire run, and DISCARDS 1 is generated as a LOAD parameter. JCL that is generated minimally sizes data sets for SYSDS001 and SYSER001 DDs. If any records must be discarded, this action causes the load utility to terminate with a return code of 8.

If the DISCARDS option is set to any value other than 1, a different discard DD (//SYSDnnnn) is generated for each load, and DISCARDS n is generated as a LOAD parameter for each LOAD command (where n is the maximum number of discard records). This action causes the load to terminate if the discard maximum is reached. If fewer records are discarded, the discard file contains the records and execution proceeds to the next step in the worklist.

DUAL=N

*(CHANGE MANAGER version 10.1 or 11.1 only)* This option indicates whether index-controlled partitioned objects can be created. DUAL must be set to match the setting of DSNZPARM PREVENT_NEW_IXCRTL_PART. Set STOPICP=Y if you have set DSNZPARM PREVENT_NEW_IXCRTL_PART to Y.
If **DUAL=N**, you can create index-controlled partitioned objects.

If **DUAL=Y**, you are not allowed to create index-controlled partitioned objects.

**DYNCOPY=N**

This option indicates whether the NGT Copy, NGT Recover, and IBM COPY utilities dynamically allocate copy (SYSCOPY) data sets (Y or N). If you use CHANGE MANAGER as a component of the Database Administration for DB2, BMC Next Generation Technology Database Administration for DB2, or BMC Object Administration for DB2 solution, the default is Y.

**DYNREORG=Y**

This option indicates whether the REORG PLUS and IBM REORG utilities dynamically allocate archive (ARCHDDN), copy (COPYDDN, RECOVERYDDN), discard (DISCARDDDN), punch (PUNCHDDN), sysrec (UNLDDN), and sortout or sysut (WORKDDN) data sets (Y or N).

**DYNUNLD=N**

This option indicates whether the UNLOAD PLUS and LOADPLUS utilities dynamically allocate unload (SYSREC) and discard (SYSDISC) data sets (Y or N). If you use the CHANGE MANAGER as a component of the Database Administration for DB2, or the BMC Object Administration for DB2 solution, the default is Y.

**EAP=AEXvrHA**

This option defines the Execution Authorization plan name, which determines if a user is authorized to run Execution.

**EIP=BMIINSTL**

This option defines the Installation plan name.

**ENVP=ACMEvryE**

This option defines the Environment plan name, which is used to display the product environment information.

**EPP=AEXvrHM**

This option defines the Execution primary plan name.

**EURO=N**

This option instructs the product to expect numbers in the European format (comma used for the decimal point) and to create output in European decimal format (Y or N).
This option is particularly important when the product parses index LIMITKEY values that are separated by commas. If the EURO keyword is present, the product requires delimiting commas to be followed by blanks.

**Note**
The Import, Specification, Baseline (CHANGE MANAGER), and Compare (CHANGE MANAGER) components use the value for EURO from the installation options module, but do not support use of the EURO keyword in the ALUIN input data stream.

**EXCEPT=nnnn**

This option is used by the product to specify the maximum number of rows that contain referential integrity or check constraint errors (exceptions). The parameter *nnnn* specifies the number of exceptions in a range from 0 to 9999. EXCEPT=0 means that no maximum exceptions exists.

**Note**
This option is not generated in the installation options module. To change the default value of the option, you must manually add the option and value to the module.

**FEP=ACMvryF**

This option defines the Front End plan name.

**GENOBID=N**

This option specifies whether to generate the OBID parameter in a CREATE TABLE statement (Y or N).

**HSMVOL=vol**

This option specifies the volume ID that indicates an archived data set if you are using a storage management system, such as the IBM Hierarchical Storage Manager (DFSMShsm). If this volume ID is encountered, the product uses a template of default values for data set allocation.

**IMP=ACMvryI**

This option defines the Import plan name.

**KEYCARD=N**

This option indicates whether to take cardinality statistics for the key columns of an index when the IBM RUNSTATS utility is used to take index statistics (Y or N).
When you alter the limit key values for an index- or table-controlled partitioned object, the products must alter each partition in a specific sequence. Each altered limit key cannot exceed the value of the existing limit key of the next partition. The products sort the limit key values so that ALTER TABLE ALTER PARTITION statements can be executed in the correct sequence. The products also validate the limit key values to ensure that partition ranges do not overlap, and that the values are correct for the data types and lengths of the partitioning columns. This option indicates whether to sort and validate the limit key values.

The parameters are defined as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Sorts and validates the limit key values.</td>
</tr>
<tr>
<td>N</td>
<td>Does not sort and validate the limit key values.</td>
</tr>
<tr>
<td>I</td>
<td>Ignores any errors that the products generate when they sort and validate the limit key values.</td>
</tr>
</tbody>
</table>

This option specifies that records be logged during loads that use the IBM LOAD utility (Y or N).

This option specifies the maximum number of SYSUT temporary work data sets that the LOADPLUS or REORG PLUS utilities can use to build nonclustering indexes for a table. The range of valid values is 1 to 9999.

This option indicates whether support for the MGMTCLAS parameter is required for VCAT-defined DB2 objects (Y or N).

When CHANGE MANAGER is used as a component of the Database Administration for DB2 or BMC Object Administration for DB2 solution, this option instructs Analysis to create commands that Execution can use to execute a worklist in parallel.

This option indicates whether to use the LOADPLUS, NGT Copy, NGT Recover, or REORG PLUS utility to create a partition-level image copy of a partitioned table space or index (Y or N).
PC=ACM

This option defines the product code to the product components.

PLANMGMT=N

This option specifies the value for PLANMGMT that is used when a rebind is executed.

The parameters are defined as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>No default</td>
</tr>
<tr>
<td>B</td>
<td>Basic</td>
</tr>
<tr>
<td>E</td>
<td>Extended</td>
</tr>
</tbody>
</table>

POFDS=('&&HLQ.CNTL(&POFNAME)'

This option specifies the name of the initial JCL Generation Product Options File (POF).

PRODUCT='PRODUCT NAME'

This option defines the product name.
(for example, PRODUCT='CHANGE MANAGER'.

REBLD=I

This option indicates whether to use the rebuild utility from IBM or BMC or no rebuild utility. If REBLD=N, eligible indexes are not created with DEFER YES. If a nonunique index is dropped or created in a worklist, and its parent table is not dropped or created in the worklist, the index is created with DEFER YES if REBLD=I or REBLD=B.

The parameters are defined as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Uses the IBM REBUILD INDEX utility to rebuild indexes.</td>
</tr>
<tr>
<td>B</td>
<td>Uses the NGT Recover utility. The NGT Recover utility must be installed.</td>
</tr>
<tr>
<td>N</td>
<td>Does not generate a rebuild utility command in the worklist.</td>
</tr>
</tbody>
</table>

RECVDD01=N
RECVDD02=N

These options define image copies for the BMC and IBM copy utilities. The parameters are defined as follows:
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Does not create an image copy.</td>
</tr>
<tr>
<td>C</td>
<td>Creates an image copy of the backup data set at the remote site, but does not register the image copy. The NGT Copy, REORG PLUS, or LOADPLUS utility is used to create the copy.</td>
</tr>
</tbody>
</table>
| R         | Creates an image copy of the backup data set at the remote site using one of the following utilities:  
- NGT Copy  
- REORG PLUS  
- LOADPLUS  
- IBM COPY  
The utility will register the copy. If the IBM COPY utility is used, RECVDD01 and RECVDD02 must equal R. |

REFRESH=Y

This option indicates whether to refresh the data in a materialized query table (MQT) by using the REFRESH TABLE statement (Y or N).

REGENATR=Y

When you rebind advanced triggers with long names or delimited names, you might encounter problems. This can be resolved by regenerating the trigger. This option specifies whether to rebind the trigger by using the REBIND PACKAGE (N) or using the ALTER TRIGGER name REGENERATE VERSION version (Y) statement.

Note
For advanced triggers, you must specify REBIND PACKAGE, not REBIND TRIGGER PACKAGE.

REGENNSP=N

When you change a DB2 object to which a native SQL stored procedure references, the package that is associated with the stored procedure might need to be rebound. This option specifies whether to rebind the package by using the REBIND PACKAGE command or the ALTER PROCEDURE statement with the REGENERATE clause (Y or N).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Generates the REBIND PACKAGE command.</td>
</tr>
<tr>
<td>Y</td>
<td>Generates the ALTER PROCEDURE statement with the REGENERATE clause.</td>
</tr>
</tbody>
</table>
**REORG=I**

This option indicates whether to use the REORG PLUS product, NGT Reorg product, or the IBM REORG utility to generate reorganization commands in worklists for changes that require reorganizing table spaces and indexes. Changes made to attributes such as PRIQTY, SECQTY, PCTFREE, FREEPAGE, and VOLUME (for VCAT-defined partitions) can cause placement of reorganization commands in the worklist.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>If DB2 places the object in a REORG-pending advisory (AREO*) status or in a REORG-pending restrictive (REORP) status, generates reorganization commands for the IBM REORG utility.</td>
</tr>
</tbody>
</table>
| B         | If DB2 places the object in an AREO* status or in a REORP status, generates reorganization commands for the BMC utility that you specify:  
  - **REORG PLUS**  
    By default the product uses REORG PLUS for reorganization.  
  - **NGT Reorg (only ALTER and CHANGE MANAGER version 12.1 and later)**  
    To select NGT Reorg, you must use the USENGTR keyword in the ALUIN input stream or specify the USENGTR= Y installation option.  

The utilities must be installed.

**REORGALL=Y**

This option indicates whether to reorganize a table space when a reorganization would be applicable (Y) or when the table space requires a reorganization (for example, the table space is in REORG-pending status) (N).

**REORGREF=Y**

This option indicates whether a table space should be reorganized by using the SHRLEVEL REFERENCE (Y) or SHRLEVEL NONE (N) option. This option is used when an online reorg (SHRLEVEL CHANGE) is either not applicable or not requested.

*Note*

NGT Reorg always performs an online reorganization (SHRLEVEL CHANGE) and disregards this option.

**REORGONL=N**

This option indicates whether a table space should be reorganized by using an online reorg (SHRLEVEL CHANGE) (Y or N).
**Note**

NGT Reorg always performs an online reorganization (SHRLEVEL CHANGE) and disregards this option.

**ROLEOWN=N**

This option indicates whether the object is to be created using trusted context with ROLE as the object owner (ROLEOWN=Y) or using the user with the associated user privileges as the owner (ROLEOWN=N).

If you have set (ROLEOWN=Y), worklists are processed using the AUTHSWOFF option.

**RPLUSYNC=R**

This option indicates whether the migration file should include all table spaces in the NGT Recover IMPORT command, or only table spaces that have changed since the preceding migration. This option applies to the Copy Migration feature of the following solutions:

- BMC Recovery for DB2
- BMC Object Administration for DB2
- BMC Next Generation Technology Database Administration for DB2
- Database Administration for DB2

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>Replaces all table spaces regardless of whether they have changed or not. Consider this option when migrating to a subsystem in which the data is manipulated in test environment.</td>
</tr>
<tr>
<td>A</td>
<td>Replaces only table spaces that have changed since the last time those spaces were imported. Consider this option when migrating to a query-based environment when the table spaces are read only.</td>
</tr>
</tbody>
</table>

**RPTPL=ACMvryR**

*(CHANGE MANAGER only)* This option specifies the name of the product plan that is used to generate reports.

**SEQI=050**

This option defines the sequence-number increment for worklists and CDL files.

**SPANNED=Y**

This option defines whether spanned records are used to unload and load XML or LOB columns.
SPP=ACM$vryS

This option defines the Specification plan name.

STATHIST=Y

This option specifies that the BMC BMCSTATS utility or the IBM RUNSTATS utility will update the DB2 catalog history tables with the current statistics that are being collected (Y or N).

STATS=S

This option indicates the type of statistics that are generated. The parameters are defined as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>Stand-alone—Generates either a -BMCS (BMC BMCSTATS) or an -RNST (IBM RUNSTATS) command in a worklist for the generation of statistics.</td>
</tr>
<tr>
<td>U</td>
<td>Utility—Uses a copy, load, or reorg utility to generate statistics whenever possible.</td>
</tr>
<tr>
<td>X</td>
<td>Does not generate statistics.</td>
</tr>
</tbody>
</table>

STOPCOMM=N

This option indicates whether an AT (COMMIT) command is generated in a worklist when a STOP command is created.

STOPICP=N

*(CHANGE MANAGER version 11.2 and later)* This option indicates whether index-controlled partitioned objects can be created. STOPICP must be set to match the setting of DSNZPARM PREVENT_NEW_IXCTRL_PART. Set STOPICP=Y if you have set DSNZPARM PREVENT_NEW_IXCTRL_PART to Y.

If STOPICP=N, you can create index-controlled partitioned objects.

If STOPICP=Y, you are not allowed to create index-controlled partitioned objects.

STOPLIST=creatorName

This option specifies the eight-character creator name for a migrate-type work ID that identifies a list of tables that should not be modified or analyzed. The default value is blank.
STORCLAS=N

This option indicates whether support for the STORCLAS parameter is required for VCAT-defined DB2 objects (Y or N).

SYNCPNT=parm

This option creates additional -SYNC commands in a worklist, based on the number of -SQL commands since the last -SYNC command. The variable parm specifies the maximum number of -SQL commands that can be in the worklist before a -SYNC command is created. Valid values for parm are from 0 to 99. The default value is 10.

When the value is reached, the Analysis component places an additional -SYNC command before the next -SQL command. Any -SYNC command in the worklist resets the count of -SQL commands to zero. -SYNC commands that this keyword generates are in addition to the -SYNC commands that Analysis automatically generates.

SYSTYPE=S

This option defines the installation’s character set.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Mixed</td>
</tr>
<tr>
<td>S</td>
<td>Single-byte only</td>
</tr>
</tbody>
</table>

TABLEACC=Y

This option indicates whether all tables remain accessible during execution.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Ensures that tables remain accessible during the execution of an alter-type worklist.</td>
</tr>
<tr>
<td>N</td>
<td>Prevents access to the tables in a worklist by putting objects into read-only status with START DATABASE commands in an alter-type worklist.</td>
</tr>
</tbody>
</table>

Whether TABLEACC=Y or TABLEACC=N, the SHRLEVEL NONE option is used as the default in a REORG TABLESPACE statement in a -BMCR or -REOR worklist command.

TABLEALL=N

This option specifies the STATS utility to gather information for all columns of tables.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Does not include the TABLE(ALL) parameter.</td>
</tr>
<tr>
<td>Y</td>
<td>Includes the TABLE(ALL) parameter in the -BMCS (BMC BMCSTATS) or the -RNST (IBM RUNSTATS) worklist command to gather statistics for all columns of all tables in a table space.</td>
</tr>
</tbody>
</table>

**UCPKPALL=N**

This option indicates whether the primary key constraint is included in a baseline or comparison.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Ignores the primary key constraint in a baseline or comparison when you exclude unique constraints.</td>
</tr>
<tr>
<td>Y</td>
<td>Includes the primary key constraint in a baseline or comparison when you exclude unique constraints.</td>
</tr>
</tbody>
</table>

**UNLDCOLL=N**

This option indicates the explicit column list that is required on all UNLOAD PLUSunloads (Y or N).

**UNLDEMPNT=Y**

This option indicates whether the tables that RUNSTATS indicates as empty are unloaded (Y or N).

**UPDSTATS=C**

This option specifies which statistics are updated. The parameters are defined as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>All—Updates the DASD tables and the DB2 catalog tables. BMCSTATS is selected.</td>
</tr>
<tr>
<td>B</td>
<td>BMC DASD tables—Updates only the DASD tables. BMCSTATS is selected.</td>
</tr>
<tr>
<td>C</td>
<td>DB2 Catalog—Updates only the DB2 catalog tables. IBM RUNSTATS is selected.</td>
</tr>
</tbody>
</table>

**USENGTR=N**

(version 12.1 or later) If you have specified the installation option REORG=B, this option determines which BMC reorganizational utility (NGT Reorg or REORG PLUS) is used to generate reorganization commands in worklists.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Uses NGT Reorg.</td>
</tr>
<tr>
<td>N</td>
<td>Uses REOR PLUS.</td>
</tr>
</tbody>
</table>

**UTILCOPY=N**

This option determines whether other utilities or a copy utility creates an image copy during loads.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Creates image copies by utilities other than the copy utilities, whenever possible. If the utilities cannot create a copy, a separate copy step is generated.</td>
</tr>
<tr>
<td>N</td>
<td>Generates all copies that the specific copy utility takes (either the IBM COPY utility or the NGT Copy utility) in a separate copy step</td>
</tr>
</tbody>
</table>

**VRM=vrmm**

This option indicates the version, release level, and maintenance level.

**VVALPROP=N**

This option indicates whether the product supports extended view text propagation (Y or N).

**ZPARM=DSNZPARM**

This option specifies a name for the DB2 DSNZPARM initialization parameter module.
JCL Generation product options

A keyword in the ALTER, CATALOG MANAGER, CHANGE MANAGER, and DASD MANAGER PLUS installation options modules, POFDS, specifies the product options file (POF).

The POF file is an 80-character sequential file that contains keywords and values for the JCL Generation options. The file is located in the HLQ.UBMCCNTL data set. When you install the products, only one POF is created. This POF, referred to as the initial POF, is initialized and populated with the default ISPF variables and values from the installation panels. Products that are installed at the same time share the initial POF.

For ALTER, CHANGE MANAGER, and DASD MANAGER PLUS, all of the POF keywords are included in the AJXPOFIN input stream (unless noted otherwise). These products use the keywords in the AJXPOFIN input stream in your initial POF and your user POF.

Example of product options

This topic provides an example of a product options file.

Figure 2: Product options file

```
****************************************** Top of Data ******************************************
POFDATE = 2016/06/14 10:00:00
* -------------------------------------------------------------
*    POF WRITTEN FROM VERSION:  V12.01.00
* FORMAT:
*   KEYWORD=PARM  COLUMNS 1-80.
*     VALUE - EVERYTHING AFTER THE = IS CONSIDERED THE VALUE.
*     LEADING AND TRAILING BLANKS ARE REMOVED.
*     VALUE,(R) TO INDICATE REFRESH OPTION.
*     NO SPLITTING OF VALUE ACROSS LINES. IF IT WON'T FIT ON
*     LINE WITH KEYWORD, ENTER '>' AFTER = AND PUT THE PARM
*     ON NEXT LINE.
*     BLANK LINES ARE IGNORED.
*     ASTERISK IN COLUMN 1 INDICATES THAT LINE IS A COMMENT.
* -------------------------------------------------------------
2MEGSQL = N
ACM_AMS = Y
ACM_ANALYSIS_SYSOUT = X
ACM_BASDIAG = SYSOUT
ACM_BRPTDIAG = SYSOUT
ACM_BRPTDSN = 'ACM.DB2V12.V121.BLREPS(&WORKID8)'
ACM_CDLDSN = 'ACM.DB2V12.V121.CDLBAS01(&WORKID8)'
ACM_CDLPS = 15
```

Chapter 4 JCL Generation product options 181
ACM_CDLSS = 5
ACM_CDLU = SYSDA
ACM_CMPDIAG = SYSDA
ACM_CPLCDLO = 'ACM.DB2V12.V121.CDLBAS01(&WORKID8)'
ACM_CPLDIAG = SYSDA
ACM_CPLWDSN = 'ACM.DB2V12.V121.TASKWL(&TASKID)'
ACM_CPLWDSNO = 'ACM.DB2V12.V121.WLBASE01(&WORKID8)'
ACM_DBRM1 =
ACM_DBRM2 =
ACM_DBRM3 =
ACM_DYNSORTW_NUM = 32
ACM_DYNSORTW_UNIT = SYSDA
ACM_GLID =
ACM_IBMR_MAP_REQ = Y
ACM_IMPDIAG = SYSDA
ACM_JDSN = 'ACM.DB2V12.V121.ANLYJCL(&WORKID8)'
ACM_JDSNB = 'ACM.DB2V12.V121.BLJCL(&WORKID8)'
ACM_JDSNBG = 'ACM.DB2V12.V121.BATJCL(&WORKID8)'
ACM_JDNSBR = 'ACM.DB2V12.V121.BLRJCL(&WORKID8)'
ACM_JDNS = 'ACM.DB2V12.V121.CMPJCL(&WORKID8)'
ACM_JDNSPL = 'ACM.DB2V12.V121.TASKJCL(&TASKID)'
ACM_JDNSPLO = 'ACM.DB2V12.V121.EXECJCL(&WORKID8)'
ACM_JDNSE = 'ACM.DB2V12.V121.EXECJCL(&WORKID8)'
ACM_JDNSN = 'ACM.DB2V12.V121.IMPJCL(&WORKID8)'
ACM_PARALLEL_MAXINIT = 5
ACM_PARALLEL_MININIT = 3
ACM_PARALLEL_WORKLST = N
ACM_PARALLEL_XIMGRP = XIMACM
ACM_PARALLEL_XIMPROC = XIMACM
ACM_PARALLEL_XIMSTRT = Y
ACM_PARALLEL_XIMTRCE = N
ACM_PIC = N
ACM_SDSN = SYSDA
ACM_SDSNE = SYSDA
ACM_WDSN = 'ACM.DB2V12.V121.WLBASE01(&WORKID8)'
ACM_WLORDER =
ACM_WLORDERMSG = N
ACM_WLPS = 15
ACM_WLSS = 5
ACM_WLU = SYSDA
ACTWRK_DATACLASS =
ACTWRK_DATACLASS_ALT =
ACTWRK_EXPDT =
ACTWRK_MGMTCLASS =
ACTWRK_MGMTCLASS_ALT =
ACTWRK_PIC = N
ACTWRK_PIC = N
ACTWRK_PITU = SYSDA
ACTWRK_PITU = SYSDA
ADDLOAD1 =
ADDLOAD2 =
ARCH_DATACLASS =
ARCH_DATACLASS_ALT =
ARCH_EXPDT =
ARCH_MGMTCLASS =
ARCH_MGMTCLASS_ALT =
ARCH_PITU = SYSDA
ARCH_PITU = SYSDA
ARCH_UNIT = SYSDA
ARCH_UNIT = SYSDA
ASU_XP_LOGD_DATAC =
ASU_XP_LOGD_MGMTC =
ASU_XP_LOGD_PITU = 10
ASU_XP_LOGD_SECQTY = 2
ASU_XP_LOGD_STOREC = 0
ASU_XP_LOGD_UNIT = SYSDA
ASU_XP_LOGDSN = &PREFIX..XPORT.LOG
ASU_XP_UIMSRVHOST =
ASU_XP_UIMSRVPORT = 1
ASU_XP_UIMSVTIMEOUT = 300
BINDFAIL = N
BLRP_DATACLASS =
BLRP_DATACLASS_ALT =
BLRP_EXPD =
BLRP_MGMTCLASS =
BLRP_MGMTCLASS_ALT =
BLRP_PREFIX = &PREFIX..&WKID..&OBNOD
BLRP_PRIQTY = 10
BLRP_RETD =
BLRP_STORCLASS =
BLRP_STORCLASS_ALT =
BLRP_THRESH = 0
BLRP_UNIT = SYSDA
BLRP_UNIT_ALT =
BMC_CHECK_LOAD =
BMC_CHECK_OPTS =
BMC_COPY_LOAD =
BMC_COPY_OPTS =
BMC_LOAD_LOAD =
BMC_LOAD_OPTS =
BMC_RECOVER_LOAD =
BMC_RECOVER_OPTS =
BMC_REORG_LOAD =
BMC_REORG_OPTS =
BMC_REORG_XBMID = XBMB
BMC_UNLOAD_LOAD =
BMC_UNLOAD_OPTS =
CAT_LOAD =
CHGMAN_LOAD =
CLEANUP_RC = 4
CNTL_DATACLASS =
CNTL_EXPD =
CNTL_MGMTCLASS =
CNTL_PREFIX = &PREFIX..&WKID..&SSID
CNTL_PRIQTY = 1
CNTL_RETD =
CNTL_SECQTY = 1
CNTL_STORCLASS =
CNTL_UNIT = SYSDA
CNTLMOUT_DSN = &PREFIX..&SSID..CNTLMOUT(&JOBNAME)
CNTLMSCH_DSN = &PREFIX..&SSID..CNTLMSCH(&JOBNAME)
CPYEXP_DATACLASS =
CPYEXP_EXPD =
CPYEXP_MGMTCLASS =
CPYEXP_PREF = &PREFIX..&WKID
CPYEXP_RETD =
CPYEXP_STORCLASS =
CPYEXP_SUPPRESS_SUFF = N
CPYEXP_UNIT = SYSDA
DASD_LOAD =
DASDDOPT = DS10QEFF
DATA_PACKER_LOAD =
DATASETSIZING = N
DATAWK_NBR = 5
DATAWK_UNIT = SYSDA
db2exit = CSG.DEJM.DSNEXIT(R)
db2load = CSG1.DB2V12M.DSNLOAD
def_gdg_base = N
def_gdg_limit = 10
def_gdg_noscr = N
def_gdg2_limit = 10
diag_msgclass = X
disc_dataclass =
disc_dataclass_alt =
disc_expd =
disc_mgmtclass =
disc_mgmtclass_alt =
disc_prefix = &PREFIX..&WKID
disc_priqty = 10
Example of product options

ALTER and CHANGE MANAGER for DB2 Reference Manual
Example of product options
Example of product options

186 ALTER and CHANGE MANAGER for DB2 Reference Manual
Example of product options
<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAPE_VOLCNT</td>
<td>99</td>
</tr>
<tr>
<td>TAPE1</td>
<td>CARTVTS</td>
</tr>
<tr>
<td>TAPE2</td>
<td>CARTVTS</td>
</tr>
<tr>
<td>TAPE3</td>
<td>CARTVTS</td>
</tr>
<tr>
<td>TEMPLATE_DSN</td>
<td></td>
</tr>
<tr>
<td>TEMPUNIT</td>
<td>SYSDA</td>
</tr>
<tr>
<td>TIMEPARM</td>
<td></td>
</tr>
<tr>
<td>TRTCH</td>
<td></td>
</tr>
<tr>
<td>TSOPROGRAM</td>
<td></td>
</tr>
<tr>
<td>TSOSUBEXIT</td>
<td>N</td>
</tr>
<tr>
<td>UNLQ</td>
<td></td>
</tr>
<tr>
<td>UNL0_DATACLASS</td>
<td></td>
</tr>
<tr>
<td>UNL0_MGMTCLASS</td>
<td></td>
</tr>
<tr>
<td>UNL0_MGMTCLASS_ALT</td>
<td></td>
</tr>
<tr>
<td>UNL0_PREFIX</td>
<td>&amp;PREFIX..&amp;MSSID..&amp;WORKID8</td>
</tr>
<tr>
<td>UNL0_PRIOTY</td>
<td>10</td>
</tr>
<tr>
<td>UNL0_SECQTY</td>
<td>2</td>
</tr>
<tr>
<td>UNL0_STORCLASS</td>
<td></td>
</tr>
<tr>
<td>UNL0_STORCLASS_ALT</td>
<td></td>
</tr>
<tr>
<td>UNL0_SUPPRESS_SUFF</td>
<td>N</td>
</tr>
<tr>
<td>UNL0_THRESH</td>
<td>0</td>
</tr>
<tr>
<td>UNL0_UNIT</td>
<td>SYSDA</td>
</tr>
<tr>
<td>UNL0_UNIT_ALT</td>
<td></td>
</tr>
<tr>
<td>UNL0_EXPDT</td>
<td></td>
</tr>
<tr>
<td>UNL02_DATACLASS</td>
<td></td>
</tr>
<tr>
<td>UNL02_MGMTCLASS</td>
<td></td>
</tr>
<tr>
<td>UNL02_MGMTCLASS_ALT</td>
<td></td>
</tr>
<tr>
<td>UNL02_PREFIX</td>
<td>&amp;PREFIX..&amp;MSSID..&amp;WORKID8</td>
</tr>
<tr>
<td>UNL02_PRIOTY</td>
<td>10</td>
</tr>
<tr>
<td>UNL02_SECQTY</td>
<td>2</td>
</tr>
<tr>
<td>UNL02_STACK</td>
<td>N</td>
</tr>
<tr>
<td>UNL02_STORCLASS</td>
<td></td>
</tr>
<tr>
<td>UNL02_STORCLASS_ALT</td>
<td></td>
</tr>
<tr>
<td>UNL02_SUPPRESS_SUFF</td>
<td>N</td>
</tr>
<tr>
<td>UNL02_THRESH</td>
<td>0</td>
</tr>
<tr>
<td>UNL02_UNIT</td>
<td>SYSDA</td>
</tr>
<tr>
<td>UNL02_UNIT_ALT</td>
<td></td>
</tr>
<tr>
<td>UNL02_EXPDT</td>
<td></td>
</tr>
<tr>
<td>UNL03_DATACLASS</td>
<td></td>
</tr>
<tr>
<td>UNL03_MGMTCLASS</td>
<td></td>
</tr>
<tr>
<td>UNL03_MGMTCLASS_ALT</td>
<td></td>
</tr>
<tr>
<td>UNL03_PREFIX</td>
<td>&amp;PREFIX..&amp;MSSID..&amp;WORKID8</td>
</tr>
<tr>
<td>UNL03_PRIOTY</td>
<td>10</td>
</tr>
<tr>
<td>UNL03_SECQTY</td>
<td>2</td>
</tr>
<tr>
<td>UNL03_STACK</td>
<td>N</td>
</tr>
<tr>
<td>UNL03_STORCLASS</td>
<td></td>
</tr>
<tr>
<td>UNL03_STORCLASS_ALT</td>
<td></td>
</tr>
<tr>
<td>UNL03_SUPPRESS_SUFF</td>
<td>N</td>
</tr>
<tr>
<td>UNL03_THRESH</td>
<td>0</td>
</tr>
<tr>
<td>UNL03_UNIT</td>
<td>SYSDA</td>
</tr>
<tr>
<td>UNL03_UNIT_ALT</td>
<td></td>
</tr>
<tr>
<td>UNL03_EXPDT</td>
<td></td>
</tr>
<tr>
<td>USE_NGT_AUTO</td>
<td>N,(R)</td>
</tr>
<tr>
<td>USER_HLD</td>
<td></td>
</tr>
<tr>
<td>USER_VAR1_CHAR</td>
<td></td>
</tr>
<tr>
<td>USER_VAR2_CHAR</td>
<td></td>
</tr>
<tr>
<td>USER_VAR3_CHAR</td>
<td></td>
</tr>
<tr>
<td>USER_VAR4_CHAR</td>
<td>0</td>
</tr>
<tr>
<td>USER_VAR5_CHAR</td>
<td>0</td>
</tr>
<tr>
<td>WORK_DATACLASS</td>
<td></td>
</tr>
<tr>
<td>WORK_MGMTCLASS</td>
<td></td>
</tr>
<tr>
<td>WORK_STORCLASS</td>
<td></td>
</tr>
</tbody>
</table>
The , (R) in the variable syntax indicates that the specified value refreshes the existing value of the variable in the user’s ISPF profile data. This update takes place when the POFDATE is later than the previous POFDATE stored in the user’s ISPF profile.

**Descriptions of product option keywords**

This topic provides descriptions of the keywords in the product options file.

See also “Example of product options” on page 181.

2MEGSQL=N

For CATALOG MANAGER, this keyword indicates whether to allocate a 2-MB buffer for large SQL statements.

This keyword is not included in the AJXPOFIN input stream.

ACM_ANALYSIS_SYSOUT =A

For ALTER and CHANGE MANAGER, this keyword sets the default output class to a value other than X.

ACM_AMS=Y

For ALTER and CHANGE MANAGER, this keyword controls whether Analysis, by default, generates AMS statements (IDCAMS DELETE and DEFINE) in the worklist. The following values are valid:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Generates AMS statements (IDCAMS DELETE and DEFINE) in a worklist</td>
</tr>
<tr>
<td>N</td>
<td>Generates a -STOP worklist command that enables you to complete the DELETE and DEFINE commands before the DB2 object CREATE commands that are located later in the worklist</td>
</tr>
</tbody>
</table>

You can use the INCLUDE (AMS) keyword to override this value.

This keyword is not included in the AJXPOFIN input stream.
**ACM_BASDIAG=SYSOUT**

For CHANGE MANAGER, this keyword sets the default value for the Baseline diagnostic output data set name.

This keyword is not included in the AJXPOFIN input stream.

**ACM_BRPTDIAG=SYSOUT**

For CHANGE MANAGER, this keyword specifies the default name for the Baseline Report diagnostic output data set.

This keyword is not included in the AJXPOFIN input stream.

**ACM_BRPTDSN='&PREFIX..BASELINE.REPORT’**

For CHANGE MANAGER, this keyword specifies the default name for the Baseline Report data set name.

This keyword is not included in the AJXPOFIN input stream.

**ACM_CDLDSN='&PREFIX..&SSID..CDL(CDL)'**

For CHANGE MANAGER, this keyword defines the default data set name for generated Change Definition Language (CDL) statements.

This keyword is not included in the AJXPOFIN input stream.

**ACM_CDLPS=15**

For CHANGE MANAGER, this keyword defines, in tracks, the default value for the primary space allocation of the CDL data set.

This keyword is not included in the AJXPOFIN input stream.

**ACM_CDLSS=5**

For CHANGE MANAGER, this keyword defines, in tracks, the default value for the secondary space allocation of the CDL data set.

This keyword is not included in the AJXPOFIN input stream.

**ACM_CDLU=SYSDA**

For CHANGE MANAGER, this keyword defines the default unit for the CDL data set.

This keyword is not included in the AJXPOFIN input stream.
ACM_CMPDIAG=SYSOUT

For CHANGE MANAGER, this keyword defines the default value for the Compare diagnostic output data set.

This keyword is not included in the AJXPOFIN input stream.

ACM_CPLCDLO=’&PREFIX..&SSID..CDL(CDL)’

For the CM/PILOT component of CHANGE MANAGER, this keyword defines the default data set name for generated CDL statements.

This keyword is not included in the AJXPOFIN input stream.

ACM_CPLDIAG=SYSOUT

For CHANGE MANAGER, this keyword specifies the default name for the CM/PILOT component’s diagnostic output data set.

This keyword is not included in the AJXPOFIN input stream.

ACM_CPLWDSN=’&PREFIX..&SSID..&TASKID’

For CHANGE MANAGER, this keyword specifies the default worklist data set name for a new TASKID used in the CM/PILOT component. CHANGE MANAGER dynamically allocates the data set the first time that the data set is used. This data set can be either a sequential file or a partitioned data set (PDS).

This keyword is not included in the AJXPOFIN input stream.

ACM_CPLWDSNO=’&PREFIX..&SSID..&WORKID’

For the CM/PILOT component of CHANGE MANAGER, this keyword specifies the default worklist data set name for a work ID. CHANGE MANAGER dynamically allocates the data set the first time that the data set is used. This data set can be either a sequential file or a partitioned data set (PDS).

This keyword is not included in the AJXPOFIN input stream.

ACM_DBRM1
ACM_DBRM2
ACM_DBRM3

For ALTER and CHANGE MANAGER, these keywords specify the name of a default DBRM library.

This keyword is not included in the AJXPOFIN input stream.
ACM_DYNSORTW_NUM = 32

For ALTER and CHANGE MANAGER, this keyword specifies the number of dynamically allocated sortwork data sets that the BMC REORG PLUS or IBM REORG utility uses.

ACM_DYNSORTW_UNIT = SYSDA

For ALTER and CHANGE MANAGER this keyword specifies the unit for dynamically allocated sortwork data sets. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

ACM_GLID= ID

For ALTER and CHANGE MANAGER, this keyword defines a global authorization ID (GLID). This authorization ID is used instead of the authorization ID of the person who submits the Execution job. The worklist begins with a -GLID command that switches authorization to the GLID.

This keyword is not included in the AJXPOFIN input stream.

ACM_IBMR_MAP_REQ=Y

For ALTER and CHANGE MANAGER, this keyword indicates whether to include the name of the mapping table in the syntax for the IBM REORG utility. The IBM REORG utility uses the mapping table to map the row IDs (RIDs) in the source table the RIDs in the target table.

Note
The REORG PLUS utility invokes the IBM DSNUTILB utility control program to enable certain features. If you have specified to use the REORG PLUS utility, you still need to specify mapping table information. For information about the features for which REORG PLUS invokes DSNUTILB, see the REORG PLUS for DB2 Reference Manual.

ACM_IMPDIAG=SYSOUT

For ALTER and CHANGE MANAGER, this keyword defines the default name for the Import diagnostic output data set.

This keyword is not included in the AJXPOFIN input stream.

ACM_JDSN='&PREFIX..ANALYSIS(&WORKID)'

For ALTER and CHANGE MANAGER, this keyword defines the default data set name that is used for Analysis JCL. This data set can be either a sequential or a partitioned data set. Hardcoding a member name is not recommended for a partitioned data set. The products automatically use the work ID as the member name.
This keyword is not included in the AJXPOFIN input stream.

**ACM_JDSNB=’&PREFIX..BASELINE(&WORKID)’**

For CHANGE MANAGER, this keyword defines the default data set name that is used for Baseline JCL. This data set can be either a sequential or a partitioned data set. Hardcoding a member name is not recommended for a partitioned data set. CHANGE MANAGER automatically uses the work ID as the member name.

This keyword is not included in the AJXPOFIN input stream.

**ACM_JDSNBR=’&PREFIX..BASELINE(BLRPTJCL)’**

For CHANGE MANAGER, this keyword specifies the default data set name where the product places the generated Baseline Report JCL. This data set can be either a sequential or partitioned data set.

This keyword is not included in the AJXPOFIN input stream.

**ACM_JDSNC=’&PREFIX..COMPARE(CMPJCL)’**

For CHANGE MANAGER, this keyword defines the default data set name that is used for Compare JCL. This data set can be either a sequential or partitioned data set.

This keyword is not included in the AJXPOFIN input stream.

**ACM_JDSNCPL=’&PREFIX..TASKID(&TASKID)’**

For CHANGE MANAGER, this keyword specifies the default data set name where the CM/PILOT component places the generated Execution JCL. This data set can be either a sequential or partitioned data set. Hardcoding a member name is not recommended for a partitioned data set. CM/PILOT automatically uses the task ID as the member name.

This keyword is not included in the AJXPOFIN input stream.
ACM_JDSNCPLO='&PREFIX..EXEC(&WORKID)'

For the CM/PILOT component of CHANGE MANAGER, this keyword defines the default data set name that is used for Execution JCL. This data set can be either a sequential or partitioned data set. Hardcoding a member name is not recommended for a partitioned data set. The product automatically uses the work ID as the member name.

This keyword is not included in the AJXPOFIN input stream.

ACM_JDSNE='&PREFIX..EXEC(&WORKID)'

For ALTER and CHANGE MANAGER, this keyword defines the default data set name that is used for Execution JCL. This data set can be either a sequential or partitioned data set. Hardcoding a member name is not recommended for a partitioned data set. The products automatically use the work ID as the member name.

This keyword is not included in the AJXPOFIN input stream.

ACM_JDSNI='&PREFIX..IMPORT(&WORKID)'

For ALTER and CHANGE MANAGER, this keyword defines the default data set name that is used for Import JCL. This data set can be either a sequential or partitioned data set. Hardcoding a member name is not recommended for a partitioned data set. The products automatically use the work ID as the member name.

This keyword is not included in the AJXPOFIN input stream.

ACM_PARALLEL_MAXINIT=3

For the Database Administration or BMC Object Administration for DB2 solution, this keyword specifies the maximum number of BMC Cross-System Image Manager (XIM) initiators to use when executing a worklist in parallel. This value controls the number of permanent work data sets that are allocated in the execution JCL. The valid range of values is 1 through 32. The maximum number of initiators should not exceed the number of objects in a worklist.

ACM_PARALLEL_MININIT=2

For the Database Administration or BMC Object Administration for DB2 solution, this keyword specifies the minimum number of XIM initiators to use when executing a worklist in parallel. If the minimum number of XIM initiators is not available, the worklist does not run. The valid range of values is 1 through 8.
ACM_PARALLEL_WORKLST=N

For the Database Administration or BMC Object Administration for DB2 solution, this keyword indicates whether a CHANGE MANAGER worklist should be executed in parallel.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Executes the worklist in parallel. However, if adequate XIM resources are not available, the Execution function fails. In addition, if the required parallelism worklist commands (such as -BEGG and -ENDG) are not included in the worklist, the worklist is not executed in parallel.</td>
</tr>
<tr>
<td>N</td>
<td>Executes the worklist sequentially, even if the required parallelism worklist commands are included in the worklist</td>
</tr>
</tbody>
</table>

ACM_PARALLEL_XIMGRP=XIMACM

For the Database Administration or BMC Object Administration for DB2 solution, this keyword specifies the group name for the XIM technology. The group name for XIM must be unique for each instance of XIM that is running on an IBM OS/390 or z/OS image.

ACM_PARALLEL_XIMPROC=XIMACM

For the Database Administration or BMC Object Administration for DB2 solution, this keyword specifies the name of the procedure that the solution uses to start the XIM technology automatically. BMC recommends that the name of the XIM started task procedure be unique for each instance of XIM that is running on an OS/390 or z/OS image.

ACM_PARALLEL_XIMSTRT=N

For the Database Administration or BMC Object Administration for DB2 solution, this keyword indicates whether to start the XIM technology automatically.

ACM_PARALLEL_XIMTRCE=N

For the Database Administration or BMC Object Administration for DB2 solution, this keyword indicates whether to use tracing while executing a worklist.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Writes tracing records to the AEXPTRAC output data set. AEXPTRAC is dynamically allocated and the output is written to SYSOUT.</td>
</tr>
<tr>
<td>N</td>
<td>Does not use tracing, even if an //AEXPTRAC DD statement is specified in the JCL</td>
</tr>
</tbody>
</table>
ACM_PIC=N

For ALTER and CHANGE MANAGER, this keyword indicates whether an image copy should be taken of each table space before a database is dropped, a table is dropped, or the table space is dropped or reorganized.

This keyword is not included in the AJXPOFIN input stream.

ACM_SDSN=SYSOUT

For ALTER and CHANGE MANAGER, this keyword specifies the default data set for diagnostic messages for Analysis. The value can be a sequential file, the keyword SYSOUT, or TERM (terminal). If you use SYSOUT, the diagnostic messages are written to the JES SPOOL. If you use TERM, the diagnostic messages are written to your terminal.

This keyword is not included in the AJXPOFIN input stream.

ACM_SDSNE=SYSOUT

For ALTER and CHANGE MANAGER, this keyword specifies the default data set for diagnostic messages for Execution. The value can be a sequential file or the keyword SYSOUT. If you use SYSOUT, the diagnostic messages are written to the JES SPOOL.

This keyword is not included in the AJXPOFIN input stream.

ACM_WDSN='&PREFIX..&SSID..&WORKID'

For ALTER and CHANGE MANAGER, this keyword defines the default data set name for a worklist that Analysis generates.

This keyword is not included in the AJXPOFIN input stream.

ACM_WLORDER

For ALTER and CHANGE MANAGER, this keyword specifies how the Analysis component sorts objects in a worklist.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Sorts the worklist by each table’s cardinality in descending sequence</td>
</tr>
<tr>
<td>N</td>
<td>Sorts the worklist by table order in ascending sequence, according to the table owner and table name</td>
</tr>
<tr>
<td>A</td>
<td>Sorts the worklist by either table cardinality or by table order, depending on whether the worklist is processed in parallel</td>
</tr>
<tr>
<td></td>
<td>If the Database Administration or BMC Object Administration for DB2 solution processes the worklist in parallel, Analysis sorts the worklist by table cardinality. Otherwise, it sorts the worklist by table order.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Blank</td>
<td>Generates the objects in the worklist in an unsorted, random order</td>
</tr>
</tbody>
</table>

This keyword is not included in the AJXPOFIN input stream.

**ACM_WLORDERMSG=Y**

For ALTER and CHANGE MANAGER, this keyword specifies whether to record in the SYSPRINT data set and in the worklist the amount of time to sort a worklist.

This keyword is not included in the AJXPOFIN input stream.

**ACM_WLPS=15**

For ALTER and CHANGE MANAGER, this keyword defines, in tracks, the default primary space allocation for the worklist.

This keyword is not included in the AJXPOFIN input stream.

**ACM_WLSS=5**

For ALTER and CHANGE MANAGER, this keyword defines, in tracks, the default secondary space allocation for the worklist.

This keyword is not included in the AJXPOFIN input stream.

**ACM_WLU=SYSDA**

For ALTER and CHANGE MANAGER, this keyword defines the default worklist unit.

This keyword is not included in the AJXPOFIN input stream.

**ACTWRK_MGMTCLASS**

For CATALOG MANAGER, this keyword specifies the SMS definition for the management class associated with the work data sets.

**ACTWRK_STORCLASS**

For CATALOG MANAGER, this keyword specifies the SMS definition for the storage class associated with the work data sets.

**ACTWRK_DATACLASS**

For CATALOG MANAGER, this keyword specifies the SMS definition for the data class associated with the work data sets.
**ACTWRK_UNIT**

For CATALOG MANAGER, this keyword specifies the unit for the work data sets. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the ACTWRK_UNIT parameter from the JCL, specify NONE.

**ACTWRK_PRIQTY**

For CATALOG MANAGER, this keyword specifies the primary allocation (in cylinders) for the work data sets if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

**ACTWRK_PRIQTY**

For CATALOG MANAGER, this keyword specifies the secondary allocation (in cylinders) for the work data sets if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

**ADDLOAD1**

This keyword defines the additional LINK library.

*Tip*

To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

**ADDLOAD2= HLQ.UBMCLINK**

This keyword defines the override LINK library. This LINK library is placed first in any STEPLIB concatenation.

*Tip*

If you are using a runtime environment, you can indicate the data set name for a different SSID by appending the &SSID or &MSSID symbolic variable to the name.

**ARCH_DATACLASS**

This keyword specifies the IBM Storage Management Subsystem (SMS) definition for the data class associated with the archive data set.

**ARCH_DATACLASS_ALT**

This keyword specifies the SMS definition for the data class associated with the archive data set (used if the threshold is exceeded).
ARCH_EXPDT

This keyword specifies the expiration date of the archive data set on tape. A data set cannot have an expiration date and a retention period. The valid formats are yyddd or yyyy/ddd.

ARCH_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the archive data set.

ARCH_MGMTCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the archive data set (used if the threshold is exceeded).

ARCH_PREFIX=&PREFIX..&WKID

This keyword specifies the prefix for the name of the archive discard data set.

ARCH_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the archive discard data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

ARCH_RETPD

This keyword specifies the retention period for the archive data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

ARCH_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the archive discard data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

ARCH_STACK=N

This keyword specifies whether to stack the archive data set on a tape with data sets of the same type (Y or N).

ARCH_STORCLASS

This keyword displays the SMS definition for the storage class associated with the archive data set.
**ARCH_STORCLASS_ALT**

This keyword specifies the SMS definition for the storage class associated with the archive data set (used if the threshold is exceeded).

**ARCH_THRESH=0**

This keyword specifies the maximum anticipated size for the archive data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

**ARCH_UNIT=SYSDA**

This keyword specifies the unit for the archive discard data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

**ARCH_UNIT_ALT**

This keyword specifies the alternate unit name for the archive data set (used if the threshold value is exceeded). The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keyword. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

**ASU_XP_LOGD_DATAC**

For DASD MANAGER PLUS, this keyword specifies the SMS data class and the allocation attributes of the Export log file.

**ASU_XP_LOGD_MGMTC**

For DASD MANAGER PLUS, this keyword specifies the SMS management class that defines the migration, retention, and backup requirements of the Export log file.

**ASU_XP_LOGD_PRIQTY=10**

For DASD MANAGER PLUS, this keyword specifies the primary allocation for the Export log file.

**ASU_XP_LOGD_SECQTY=2**

For DASD MANAGER PLUS, this keyword specifies the secondary allocation for the Export log file.
ASU_XP_LOGD_STORC=10

For DASD MANAGER PLUS, this keyword specifies the SMS storage class that defines the processing requirements of the Export log file.

ASU_XP_LOGD_UNIT=SYSDA

For DASD MANAGER PLUS, this keyword specifies the unit for the Export log file.

ASU_XP_LOGD_LOGDSN=&PREFIX..XPORT.LOG

For DASD MANAGER PLUS, this keyword specifies the Export log file.

ASU_XP_UIMSRVHOST=

For DASD MANAGER PLUS, this keyword specifies the host name of the primary UIM server which contains the host definitions repository for the Export utility.

ASU_XP_UIMSRVPORT=1

For DASD MANAGER PLUS, this keyword specifies the port number of the primary UIM server that contains the host definitions repository for the Export utility.

ASU_XP_UIMSRVTIMEOUT=300

For DASD MANAGER PLUS, this keyword specifies the UIM timeout parameter that determines how long the Export utility should wait for a response from the UIM server before timing out.

BINDFAIL=N

This keyword specifies whether worklist execution continues if a bind fails:

- If BINDFAIL=Y, worklist execution stops with a return code of 8. The stop is noted in the sync tables, and an Execution restart continues with the command that caused the failure.

- If BINDFAIL=N, worklist execution continues.

BLRP_DATACLASS

For CHANGE MANAGER, this keyword specifies the SMS definition for the data class associated with the baseline recovery point data set.
BLRP_DATACLASS_ALT

For CHANGE MANAGER, this keyword specifies the SMS definition for the data class associated with the baseline recovery point data set (used if the threshold is exceeded).

BLRP_EXPDT

For CHANGE MANAGER, this keyword specifies the expiration date of the baseline recovery point data set on tape. The valid formats are yyddd or yyyy/ddd.

BLRP_MGMTCLASS

For CHANGE MANAGER, this keyword specifies the SMS definition for the storage class associated with the baseline recovery point data set.

BLRP_MGMTCLASS_ALT

For CHANGE MANAGER, this keyword specifies the SMS definition for the storage class associated with the baseline recovery point data set (used if the threshold is exceeded).

BLRP_PREFIX=&PREFIX..&OBNOD

For CHANGE MANAGER, this keyword defines the high-level qualifier, or prefix, used for data sets containing data stored for a baseline recovery point.

BLRP_PRIQTY=10

For CHANGE MANAGER, this keyword defines the primary allocation quantity for baseline recovery point data sets if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

BLRP_RETPD

For CHANGE MANAGER, this keyword specifies the retention period for the baseline recovery point data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

BLRP_SECQTY=2

For CHANGE MANAGER, this keyword defines the secondary allocation quantity for baseline recovery point data sets if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 to 99999.

BLRP_STACK=N

For CHANGE MANAGER, this keyword specifies whether to stack full-recovery baseline data sets on tape (Y or N).
BLRP_STORCLASS

For CHANGE MANAGER, this keyword specifies the SMS definition for the storage class associated with the baseline recovery point data set.

BLRP_STORCLASS_ALT

For CHANGE MANAGER, this keyword specifies the SMS definition for the storage class associated with the baseline recovery point data set (used if the threshold is exceeded).

BLRP_THRESH=0

For CHANGE MANAGER, this keyword specifies the maximum anticipated size for the baseline recovery point data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

BLRP_UNIT=SYSDA

For CHANGE MANAGER, this keyword specifies the unit used for the baseline recovery point data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

BLRP_UNIT_ALT

For CHANGE MANAGER, this keyword specifies the alternate unit name for the baseline recovery point data set (used if the threshold value is exceeded). The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

BMC_CHECK_LOAD

This keyword specifies the name of the LINK library for the CHECK PLUS utility.

Tip

To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

BMC_CHECK_OPTS=ACK$OPTS

This keyword specifies the name of the CATALOG MANAGER utility installation options module. This keyword replaces the CHECKDOPT AJXPOFIN keyword. If both BMC_CHECK_OPTS and CHECKDOPT are included in the POF, the components use the value that is specified for CHECKDOPT.
**BMC_COPY_LOAD**

This keyword specifies the name of the LINK library for the NGT Copy utility.

*Tip*

To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

**BMC_COPY_OPTS=ACP$OPTS**

This keyword specifies the name of the NGT Copy utility installation options module. This keyword replaces the COPYDOPT AJXPOFIN keyword. If both BMC_COPY_OPTS and COPYDOPT are included in the POF, the components use the value that is specified for COPYDOPT.

**BMC_LOAD_LOAD**

This keyword specifies the name of the LINK library for the BMC LOADPLUS utility.

*Tip*

To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

**BMC_LOAD_OPTS=AMU$OPTS**

This keyword specifies the name of the BMC LOADPLUS utility installation options module. This keyword replaces the LOADDOPT AJXPOFIN keyword. If both BMC_LOAD_OPTS and LOADDOPT are included in the POF, the components use the value that is specified for LOADDOPT.

**BMC_RECOVER_LOAD**

This keyword specifies the name of the LINK library for the NGT Recover utility.

*Tip*

To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

**BMC_RECOVER_OPTS=AFR$OPTS**

This keyword specifies the name of the installation options module for the NGT Recover utility. This keyword replaces the RECOVERDOPT AJXPOFIN keyword. If both BMC_RECOVER_OPTS and RECOVERDOPT are included in the POF, the components use the value that is specified for RECOVERDOPT.
**BMC_REORG_LOAD**

This keyword specifies the name of the LINK library for the BMC REORG PLUS utility.

---

**Tip**

To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

---

**BMC_REORG_OPTS=ARU$OPTS**

This keyword specifies the name of the installation options module for the BMC REORG PLUS utility. This keyword replaces the REORGDOPT AJXPOFIN keyword. If both BMC_REORG_OPTS and REORGDOPT are included in the POF, the components use the value that is specified for REORGDOPT.

---

**BMC_REORG_XBMID**

This keyword specifies the BMC EXTENDED BUFFER MANAGER (XBM) subsystem (SSID) that the BMC REORG PLUS utility accesses when it uses XBM or the XBM SNAPSHOT UPGRADE FEATURE (SUF) to create a snapshot of the data sets to be reorganized.

ALTER and CHANGE MANAGER use this value when they reorganize a table space by using an online reorg (SHRLEVEL CHANGE). CATALOG MANAGER and DASD MANAGER PLUS do not use this value. The value of the SSID can be from 1 to 8 characters long.

---

**BMC_UNLOAD_LOAD**

This keyword specifies the name of the LINK library for the BMC UNLOAD PLUS utility.

---

**Tip**

To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

---

**BMC_UNLOAD_OPTS=ADU$OPTS**

This keyword specifies the name of the installation options module for the BMC UNLOAD PLUS utility. This keyword replaces the UNLOADDOPT AJXPOFIN keyword. If both BMC_UNLOAD_OPTS and UNLOADDOPT are included in the POF, the components use the value that is specified for UNLOADDOPT.

---

**CAT_LOAD**

This keyword specifies the name of the LINK library for the BMC CATALOG MANAGER product.
Tip
To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

CHECKDOPT=ACK$MMS

This keyword specifies the name of the installation options module for the CATALOG MANAGER utility. The BMC_CHECK_OPTS AJXPOFIN keyword replaces this keyword. If both BMC_CHECK_OPTS and CHECKDOPT are included in the POF, the components use the value that is specified for CHECKDOPT.

CHGMAN_LOAD

This keyword specifies the name of the LINK library for CHANGE MANAGER.

Tip
To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

CLEANUP_RC=4

This keyword specifies the value of the return code from the JCL cleanup job step. The cleanup job step, which deletes permanent work data sets, is only performed if the condition code that is returned from any previous job step is less than or equal to the code specified in CLEANUP_RC.

CNTL_DATACLASS

This keyword specifies the SMS definition for the data class associated with the control data set.

CNTL_EXPDT

This keyword specifies the expiration date of the control data set on tape. A data set cannot have an expiration date and a retention period. The valid formats are yyddd or yyyy/ddd.

CNTL_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the control data set.

CNTL_PREFIX=&PREFIX..&WKID..&SSID

This keyword specifies the prefix for the name of the control data set that the BMC UNLOAD PLUS utility uses.
CNTL_PRIQTY=1

This keyword specifies the primary allocation (in cylinders) for the control
data set if DATASETSIZING=N or if an error in sizing occurs. Valid values
are 1 through 99999.

CNTL RETPD

This keyword specifies the retention period for the control data set on tape. A
data set cannot have an expiration date and a retention period. Valid values
are 1 through 9999.

CNTL_SECQTY=1

This keyword specifies the secondary allocation (in cylinders) for the control
data set if DATASETSIZING=N or if an error in sizing occurs. Valid values
are 1 to 9999.

CNTL_STORCLASS

This keyword specifies the SMS definition for the storage class associated
with the control data set.

CNTL UNIT=SYSDA

This keyword specifies the unit name for the control data set. The value of
the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit
the UNIT parameter from the JCL, specify NONE.

CNTLMOUT_DSN=&PREFIX..&SSID..CNTLMOUT(&JOBNAME)

For DASD MANAGER PLUS, this keyword specifies the name of the output
data set that contains the job schedule name and the job sequence number of
the non-IEFBR14 jobs.

CNTLMSCH_DSN=&PREFIX..&SSID..CNTLMSCH(&JOBNAME)

For DASD MANAGER PLUS, this keyword specifies the name of the input
data set that contains the BMC Control-M job schedule.

COPYDOPT=ACP$MMS

This keyword specifies the name of the installation options module for the
NGT Copy utility. The BMC_COPY_OPTS AJXPOFIN keyword replaces this
keyword. If both BMC_COPY_OPTS and COPYDOPT are included in the
POF, the components use the value that is specified for COPYDOPT.
CPYEXP_DATACLASS

This keyword specifies the SMS definition for the data class associated with the EXPORT data set that the NGT Copy EXPORT command creates to migrate data.

CPYEXP_EXPDT

This keyword specifies the expiration date of the EXPORT data set on tape that the NGT Copy EXPORT command creates to migrate data. A data set cannot have an expiration date and a retention period. The valid formats are yyddd or yyyy/ddd.

CPYEXP_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the EXPORT data set that the NGT Copy EXPORT command creates to migrate data.

CPYEXP_PREFIX=&PREFIX..&WKID

This keyword specifies the prefix for the name of the EXPORT data set that the NGT Copy EXPORT command creates to migrate data.

CPYEXP_RETPD

This keyword specifies the retention period for the EXPORT data set on tape that the NGT Copy EXPORT command creates to migrate data. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

CPYEXP_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the EXPORT data set that the NGT Copy EXPORT command creates to migrate data.

CPYEXP_SUPPRESS_SUFF=N

This keyword specifies whether to suppress adding the DD name to the end of the name of the EXPORT data set that the NGT Copy EXPORT command creates to migrate data (Y or N). If you specify Y, you must ensure that the data set name is unique.

CPYEXP_UNIT=SYSDA

This keyword specifies the unit name for the EXPORT data set that the NGT Copy EXPORT command creates to migrate data. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.
DASD_LOAD

This keyword specifies the name of the LINK library for the BMC DASD MANAGER PLUS product.

Tip
To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

DASDDOPT

This keyword specifies the name of the installation options module for the BMC DASD MANAGER PLUS product. ALTER and CHANGE MANAGER can use the BMCSTATS utility to collect statistics and populate the DASD MANAGER PLUS database, which is maintained in DB2 tables. If you select to use the BMCSTATS utility, you must specify a value for this keyword.

DATA_PACKER_LOAD

This keyword specifies the name of the LINK library for the BMC DATA PACKER product.

Tip
To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

DATASETSIZING=N

This keyword specifies the type of data set sizing. The following values are valid:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Does not perform data set sizing</td>
</tr>
<tr>
<td>C</td>
<td>Uses IBM RUNSTATS to perform data set sizing by using statistics from the DB2 catalog</td>
</tr>
<tr>
<td>B</td>
<td>Uses BMCSTATS to perform data set sizing by using the statistics from the BMC DASD MANAGER PLUS product tables</td>
</tr>
<tr>
<td>O</td>
<td>Physically and randomly samples the VSAM objects to estimate data set sizes</td>
</tr>
</tbody>
</table>

DATAWK_NBR=4

For CATALOG MANAGER, this keyword specifies the number of DATAWK data sets that the IBM REORG utility conditionally uses for sorting data.
DATAWK_UNIT=SYSDA

For CATALOG MANAGER, this keyword specifies the unit name of the DATAWK data set that the IBM REORG utility conditionally uses for sorting data. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

DB2EXIT

This keyword specifies the name of the DB2 EXIT library.

Tip
To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

DB2LOAD

This keyword specifies the name of the DB2 LOAD library.

Tip
To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

DEF_GDG_BASE=N

This keyword specifies whether to create the base of the generation data group (GDG) at JCL generation time (Y or N).

DEF_GDG_LIMIT=10

This keyword specifies the maximum number of GDG data sets that are allowed for primary copies. Valid values are 1 through 255.

DEF_GDG_NOSCR=N

This keyword specifies whether the base of a generation data group (GDG) is defined in the IDCAMS DEFINE command as EMPTY (NSCR). If the GDG is defined as EMPTY (NSCR), the operating system uncatalogs the generation data set when it reaches the maximum number of generation data sets to keep (LIMIT). Otherwise, if the GDG is defined as SCRATCH (SCR), the operating system scratches (deletes) the generation data set when the data set is uncataloged.

DEF_GDG2_LIMIT=10

This keyword specifies the maximum number of GDG data sets that are allowed for recovery copies. Valid values are 1 through 255.
DIAG_MSGCLASS

This keyword specifies the SYSOUT class that the components use for reporting incorrect entries in the POF. The default value is blank, which indicates that a report is not generated when the product is invoked. The asterisk (*) is a valid value in batch mode.

DISC_DATACLASS

This keyword specifies the SMS definition for the discard data set’s data class.

DISC_DATACLASS_ALT

This keyword specifies the SMS definition for the discard data set’s data class (used if the threshold is exceeded).

DISC_EXPDT

This keyword specifies the expiration date of the discard data set on tape. A data set cannot have an expiration date and a retention period. The valid formats are yyddd or yyyy/ddd.

DISC_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the discard data set.

DISC_MGMTCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the discard data set (used if the threshold is exceeded).

DISC_PREFIX=&PREFIX..&OBNOD

This keyword specifies the prefix for the name of the discard data set.

DISC_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the discard data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

DISC_RETPD

This keyword specifies the retention period for the discard data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.
DISC_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the discard data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

DISC_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the discard data set.

DISC_STORCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the discard data set (used if the threshold is exceeded).

DISC_THRESH=0

This keyword specifies the maximum anticipated size for the discard data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

DISC_UNIT=SYSDA

This keyword specifies the unit for the discard data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

DISC_UNIT_ALT

This keyword specifies the alternate unit name for the discard data set (used if the threshold value is exceeded). The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

DISP_ALLOW_POPUP=N

For ALTER and CHANGE MANAGER, this keyword specifies whether to display a dialog or a panel when the ZOOM (F4) key is pressed on an object name.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Indicates to display the object name in a dialog</td>
</tr>
<tr>
<td></td>
<td>If the name is too long to be displayed in a dialog, the product displays</td>
</tr>
<tr>
<td></td>
<td>the name in a panel</td>
</tr>
<tr>
<td>N</td>
<td>Indicates to display the object name in a panel</td>
</tr>
</tbody>
</table>

212   ALTER and CHANGE MANAGER for DB2 Reference Manual
DISP_AUTO_TAB=

For ALTER and CHANGE MANAGER, this keyword specifies whether to display an autotab character in front of an object name that is too long to be displayed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Indicates not to display an autotab character</td>
</tr>
<tr>
<td>any character other than N</td>
<td>Displays as the autotab character</td>
</tr>
</tbody>
</table>

DISP_LOCATION=M

For ALTER, CHANGE MANAGER, and DASD MANAGER PLUS, this keyword specifies the location of characters to be omitted in an object name that is too long to be displayed.

The following values are valid:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>Replaces characters at the left end (beginning) of the name</td>
</tr>
<tr>
<td>M</td>
<td>Replaces characters in the middle of the name</td>
</tr>
<tr>
<td>E</td>
<td>Replaces characters at the right end (end) of the name</td>
</tr>
</tbody>
</table>

DISP_OMIT_CHAR= < >

For ALTER, CHANGE MANAGER, and DASD MANAGER PLUS, this keyword specifies the characters that replace the beginning and end of a truncated string in an object name that is too long to be displayed.

DISP_STATS=N

This keyword specifies whether to include comments that show the statistics that the components use to determine the sizes of the data sets in the generated JCL (Y or N).

DISP_VAR_DEBUG=N

This keyword specifies whether to include the SLIB variables that JCL Generation uses to create the JCL, as well as their assigned values, in the generated JCL (Y or N).

DROPR_NOIC=N

For ALTER, CATALOG MANAGER, and CHANGE MANAGER, this keyword specifies whether to invoke the Drop Recovery feature of the BMC CATALOG MANAGER product and drop an object (Y or N).
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Does not allow an object to be dropped, if no image copies of the object exist</td>
</tr>
<tr>
<td>Y</td>
<td>Allows an object to be dropped, even if no image copies of the object exist</td>
</tr>
</tbody>
</table>

**DSNCHECK44=N**

This keyword specifies whether to verify that the prefix of a data set name contains 44 characters (Y or N). Typically, JCL Generation verifies whether a prefix of a data set name contains 35 characters.

Change the value of this keyword to Y for the following reasons:
- You are modifying an SLIB because the ddnames that BMC generated do not meet your environment’s standards.
- You are creating the name of a data set.
- The data set name does not refer to a generation data group (GDG).

If you set the value to Y, you must ensure that the data set names are unique.

**DSNTIAD_PLAN**

This keyword specifies the name of the DB2 plan to run the IBM DSNTIAD program.

**ERR_DATACLASS**

This keyword specifies the SMS definition for the data class associated with the error data set.

**ERR_DATACLASS_ALT**

This keyword specifies the SMS definition for the data class associated with the error data set (used if the threshold is exceeded).

**ERR_EXPDT**

This keyword specifies the expiration date of the error data set on tape. A data set cannot have an expiration date and a retention period. The valid formats are yyddd or yyyy/ddd.

**ERR_MGMTCLASS**

This keyword specifies the SMS definition for the storage class associated with the error data set.

**ERR_MGMTCLASS_ALT**

This keyword specifies the SMS definition for the storage class associated with the error data set (used if the threshold is exceeded).
ERR_PREFIX=&PREFIX..&WKID..&STEPN

This keyword specifies the prefix for the name of the error data set.

ERR_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the error data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

ERR_RETPD

This keyword specifies the retention period for the error data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

ERR_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the error data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

ERR_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the error data set.

ERR_STORCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the error data set (used if the threshold is exceeded).

ERR_THRESH=0

This keyword specifies the maximum anticipated size for the error data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

ERR_UNIT=SYSDA

This keyword specifies the unit for the error data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

ERR_UNIT_ALT

This keyword specifies the alternate unit name for the error data set (used if the threshold value is exceeded). The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name
from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

EXEC_LOAD

This keyword specifies the name of the Execution LINK library.

Tip

To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

FCPY_DATACLASS

For CATALOG MANAGER, this keyword specifies the SMS definition for the data class associated with the flashcopy data set. The IBM FlashCopy feature creates the flashcopy (a point-in-time copy of a volume).

FCPY_EXPDT

For CATALOG MANAGER, this keyword specifies the expiration date of the flashcopy copy data set on tape. The IBM FlashCopy feature creates the flashcopy (a point-in-time copy of a volume). A data set cannot have an expiration date and a retention period. The valid formats are yyddd or yyyy/ddd.

FCPY_MGMTCLASS

For CATALOG MANAGER, this keyword specifies the SMS definition for the management class associated with the flashcopy data set. The IBM FlashCopy feature creates the flashcopy (a point-in-time copy of a volume).

FCPY_PREFIX=&PREFIX..&OBNOD..P&PART

For CATALOG MANAGER, this keyword specifies the prefix for the name of the flashcopy data set. The IBM FlashCopy feature creates the flashcopy (a point-in-time copy of a volume).

FCPY_PRIQTY=10

For CATALOG MANAGER, this keyword specifies the primary allocation (in cylinders) for the flashcopy data set if DATASETSIZING=N or if an error in sizing occurs. The IBM FlashCopy feature creates the flashcopy (a point-in-time copy of a volume). Valid values are 1 through 99999.

FCPY_RETPD

For CATALOG MANAGER, this keyword specifies the retention period for the flashcopy data set on tape. The IBM FlashCopy feature creates the flashcopy (a point-in-time copy of a volume). A data set cannot have an expiration date and a retention period. Valid values are 1 to 9999.
FCPY_SECQTY=2

For CATALOG MANAGER, this keyword specifies the secondary allocation (in cylinders) for the flashcopy data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999. The IBM FlashCopy feature creates the flashcopy (a point-in-time copy of a volume).

FCPY_STORCLASS

For CATALOG MANAGER, this keyword specifies the SMS definition for the storage class associated with the flashcopy data set. The IBM FlashCopy feature creates the flashcopy (a point-in-time copy of a volume).

FCPY_SUPPRESS_SUFF=N

For CATALOG MANAGER, this keyword specifies whether to suppress adding the DD name to the end of the name of the flashcopy data set (Y or N). If you specify Y, you must ensure that the data set name is unique. The IBM FlashCopy feature creates the flashcopy (a point-in-time copy of a volume).

FCPY_UNIT=SYSDA

For CATALOG MANAGER, this keyword specifies the unit for the flashcopy data set. The IBM FlashCopy feature creates the flashcopy (a point-in-time copy of a volume). The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

FILT_DATACLASS

This keyword specifies the SMS definition for the data class associated with the filter data set.

FILT_EXPDT

This keyword specifies the expiration date of the filter data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyddd or yyyy/ddd.

FILT_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the filter data set.

FILT_PREFIX=&PREFIX..&WKID..&STEPN

This keyword specifies the prefix for the name of the filter data set.
FILT_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the filter data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

FILT_RETPD

This keyword specifies the retention period for the filter data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

FILT_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the filter data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

FILT_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the filter data set.

FILT_UNIT=SYSDA

This keyword specifies the unit name for the filter data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

GDG_MODEL=SYS1.MODEL

This keyword specifies the name of the GDG model data set.

Note

If GDG_MODEL= NONE, DCB=model.dataSetName is omitted from the JCL for the data set.

HASHFAIL=N

This keyword specifies whether Execution terminates a job if a hash failure, such as a changed or added statement, occurs in a worklist.

HASHWARNRC

This keyword defines the return code that the product sends back when the product finds only hash warnings.

Note

Do not use 8 for this value.
INCLUDE_SYSPRIN2=N

This keyword specifies whether to add the following DD to execution JCL for utility jobs:

//SYSPRIN2 DD SYSOUT=*  

The SYSPRIN2 output data set contains SYSPRINT output messages for versions 10.2 and later of the following BMC utilities:

- CHECK PLUS  
- LOADPLUS  
- REORG PLUS  
- UNLOAD PLUS  

If you select Y, you can view the SYSPRINT output from a utility while an execution job runs the utility or when an execution job cancels during the running of the utility.

**Note**

SYSPRIN2 data sets have the following restrictions:

- When you specify BMCSTATS YES or UPDATEDB2STATS YES for LOADPLUS or REORG PLUS, SYSPRIN2 does not contain the statistics report from the Common Statistics component.

- When invoking the IBM DSNUTILB utility, REORG PLUS and LOADPLUS ignore the SYSPRIN2 DD statement.

IOALOAD1

This keyword specifies the name of a LINK library for the utility automation component of the BMC Database Performance for DB2 solution.

**Tip**

To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

IOALOAD2

This keyword specifies the name of a LINK library for the utility automation component of the BMC Database Performance for DB2 solution.

**Tip**

To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.
**JCLCLEANUP=N**

This keyword specifies whether to generate a job step that automatically deletes many of the permanent work data sets that Execution creates (Y or N).

**JCLLIB**

This keyword specifies the name of a partitioned data set (PDS) that contains JCL to be included in a job, or the name of a PDS that specifies the cataloged procedures (PROCs) that are used for non-worklist JCL.

**JES3=N**

JCL Generation no longer uses this keyword.

**JOBINCLUDE_MEMBER**

This keyword specifies the name of a JCL member to be included at the end of a job.

**JOBCARD1=/>**

//JOBC JOB (&ZACCTNUM),+'&PGMR'

**JOBCARD2=/> CLASS=A,MSGLEVEL=(1,1)**

**JOBCARD3=/>**

**JOBCARD4=/>**

**JOBCARD5=/>**

These keywords define the default job card statement that the components use when JCL Generation generates Analysis and Execution JCL. For DASD MANAGER PLUS, these keywords specify the BMCTRIG and batch report job cards.

**LISTDEF_DSN**

For CATALOG MANAGER, this keyword specifies the name of the data set that contains member names for LISTDEF utility control statements.

**LL_CLIB=CLIB**

**LL_CLIB2**

**LL_CLIB3**

**LL_CLIB4**

**LL_CLIB5**

These keywords specify the low-level qualifier (LLQ) for the product CLIST data sets for the batch ISPF environment.
LL_LINK5

These keywords specify the LLQ for the LOAD library data sets for the batch ISPF environment.

LL_MLIB=MLIB
LL_MLIB2
LL_MLIB3
LL_MLIB4
LL_MLIB5

These keywords specify the LLQ for the message data sets for the batch ISPF environment.

LL_NAUTO=AUTO
LL_NAUTO2
LL_NAUTO3
LL_NAUTO4
LL_NAUTO5

These keywords control the behavior of JCLGEN and the NGTAUTO DD in the execution JCL when -NGTU commands exist in the worklist.

LL_PLIB=PLIB
LL_PLIB2
LL_PLIB3
LL_PLIB4
LL_PLIB5

These keywords specify the LLQ for the panel and Help library data sets for the batch ISPF environment.

LL_SLIB=SLIB
LL_SLIB2
LL_SLIB3
LL_SLIB4
LL_SLIB5

These keywords specify the LLQ for the ISPF skeleton data sets for the batch ISPF environment.

LL_TLIB=TLIB
LL_TLIB2
LL_TLIB3
LL_TLIB4
LL_TLIB5

These keywords specify the LLQ for the ISPF table data sets for the batch ISPF environment.

LL_XML=XML
LL_XML2
LL_XML3
These keywords specify the LLQ for the utility generation data sets for the batch ISPF environment.

**LLQ**

This keyword specifies the LLQ for ISPF data sets for the batch ISPF environment. During installation, if you chose to use the runtime enablement feature, the Installation System sets this value to BMC. If you chose not to use the feature, the Installation System sets the value to DB.

---

**Note**

Changing the qualifier of the ISPF data sets might cause unpredictable results. Do not change the LLQ for the ISPF data sets.

**LOADDOPT**

This keyword specifies the name of the installation options module for the BMC LOADPLUS utility. The BMC_LOAD_OPTS AJXPOFIN keyword replaces this keyword. If both BMC_LOAD_OPTS and LOADDOPT are included in the POF, the components use the value that is specified for LOADDOPT.

**LOGWK_NBR=4**

This keyword specifies the number of LOGSORT data sets. Valid values are 1 through 32.

**LOGWK_UNIT=SYSDA**

This keyword specifies the unit name of the LOGSORT data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

**MAP_DATACLASS**

This keyword specifies the SMS definition for the data class associated with the map data set.

**MAP_DATACLASS_ALT**

This keyword specifies the SMS definition for the data class associated with the map data set (used if the threshold is exceeded).

**MAP_EXPDT**

This keyword specifies the expiration date of the map data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyyd or yyyy/ddd.
MAP_MGMTCLASS
This keyword specifies the SMS definition for the storage class associated with the map data set.

MAP_MGMTCLASS_ALT
This keyword specifies the SMS definition for the storage class associated with the map data set (used if the threshold is exceeded).

MAP_PREFIX=&PREFIX..&WKID..&SSID
This keyword specifies the prefix for the name of the map data set.

MAP_PRIQTY=10
This keyword specifies the primary allocation (in cylinders) for the map data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

MAP_RETPD
This keyword specifies the retention period for the map data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

MAP_SECQTY=2
This keyword specifies the secondary allocation (in cylinders) for the map data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

MAP_STORCLASS
This keyword specifies the SMS definition for the storage class associated with the map data set.

MAP_STORCLASS_ALT
This keyword specifies the SMS definition for the storage class associated with the map data set (used if the threshold is exceeded).

MAP_THRESH=0
This keyword specifies the maximum anticipated size for the map data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.
MAP_UNIT=SYSDA

This keyword specifies the unit for the map data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

MAP_UNIT_ALT

This keyword specifies the alternate unit name for the map data set (used if the threshold value is exceeded). The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

MAX_CYL=99999

This keyword specifies the maximum number of cylinders in the data set. If this value is exceeded for a data set, MAX_PRIQTY and MAX_SECQTY are used for any type of data set that does not have a specific threshold limit specified. Valid values are 1 through 99999.

MAX_PRIQTY=2000

This keyword specifies the primary quantity in cylinders that will be used when the value of MAX_CYL is reached. Valid values are 1 through 9999.

MAX_SECQTY=200

This keyword specifies the secondary quantity in cylinders that will be used when the value of MAX_CYL is reached. Valid values are 1 through 9999.

MAX_UNITCNT

This keyword specifies the value for the DASD unit count. Valid values are 1 through 59.

MEMLIMIT

This keyword specifies the limit on above-the-bar memory for an address space.

NGT_UTILDB=defaultDbname

This keyword provides the database name for the NGT utility parameter +DBNAME(dbName). NGT Reorg creates and uses a temporary table space in this database.

NGT_UTILDB is included in the AJXPOFIN input stream for batch processing.
ORTPARM_DSN

This keyword specifies the name of the data set for the SyncSort parameters.

PCPY1_DATACLASS

This keyword specifies the SMS definition for the data class associated with the local primary copy data set.

PCPY1_DATACLASS_ALT

This keyword specifies the SMS definition for the data class associated with the local primary copy data set (used if the threshold is exceeded).

PCPY1_EXPDT

This keyword specifies the expiration date of the local primary copy data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyddd or yyyy/ddd.

PCPY1_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the local primary copy data set.

PCPY1_MGMTCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the local primary copy data set (used if the threshold is exceeded).

PCPY1_PREFIX=&PREFIX..&OBNOD..P&PART

This keyword specifies the prefix for the name of the local primary copy data set.

PCPY1_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the local primary copy data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

PCPY1_RETPD

This keyword specifies the retention period for the local primary copy data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 to 9999.
PCPY1_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the local primary copy data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

PCPY1_STACK=N

This keyword specifies whether to stack the local primary copy data set on a tape with data sets of the same type (Y or N).

PCPY1_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the local primary copy data set.

PCPY1_STORCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the local primary copy data set (used if the threshold is exceeded).

PCPY1.Suppress_SUFF=N

This keyword specifies whether to suppress adding the DD name to the end of the name of the local primary copy data set (Y or N). If you specify Y, you must ensure that the data set name is unique.

PCPY1_THRESH=0

This keyword specifies the maximum anticipated size for the local primary copy data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

PCPY1_UNIT=SYSDA

This keyword specifies the unit for the local primary copy data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

PCPY1_UNIT_ALT

This keyword specifies the alternate unit name for the local primary copy data set (used if the threshold value is exceeded). The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.
PCPY2_DATACLASS

This keyword specifies the SMS definition for the data class associated with the local backup copy data set.

PCPY2_DATACLASS_ALT

This keyword specifies the SMS definition for the data class associated with the local backup copy data set (used if the threshold is exceeded).

PCPY2_EXPDT

This keyword specifies the expiration date of the local backup copy data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyddd or yyyy/ddd.

PCPY2_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the local backup copy data set.

PCPY2_MGMTCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the local backup copy data set (used if the threshold is exceeded).

PCPY2_PREFIX=&PREFIX..&OBNOD..P&PART

This keyword specifies the prefix for the name of the local backup copy data set.

PCPY2_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the local backup copy data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

PCPY2_RETPD

This keyword specifies the retention period for the local backup copy data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

PCPY2_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the local backup copy data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.
PCPY2_STACK=N

This keyword specifies whether to stack the local backup copy data set on a tape with data sets of the same type (Y or N).

PCPY2_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the local backup copy data set.

PCPY2_STORCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the local backup copy data set (used if the threshold is exceeded).

PCPY2_SUPPRESS_SUFF=N

This keyword specifies whether to suppress adding the DD name to the end of the name of the local backup copy data set (Y or N). If you specify Y, you must ensure that the data set name is unique.

PCPY2_THRESH=0

This keyword specifies the maximum anticipated size for the local backup copy data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

PCPY2_UNIT=SYSDA

This keyword specifies the unit for the local backup copy data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

PCPY2_UNIT_ALT

This keyword specifies the alternate unit name for the local backup copy data set (used if the threshold value is exceeded). The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

POFDATE

This keyword shows the last date on which the initial product options file (POF) was updated. This value is created or updated when the POF is created or when it is updated by the AJXPOVAL or AJXPODAT edit macros.
**PRE_JOBSTEP_INCLUDE**

This keyword specifies the name of a JCL member to be included before each step in the JCL.

**PROC_BMCCHECK_NAME**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the CHECK PLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_BMCCHECK_STEP**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the CHECK PLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_BMCCOPY_NAME**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the NGT Copy utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_BMCCOPY_STEP**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the NGT Copy utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_BMCCPRS_NAME**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the BMC DASD MANAGER PLUS for DB2 utility. BMCCPRS references the utility that copies statistics from the DB2 catalog to the BMCSTATS tables.

This keyword is not included in the AJXPOFIN input stream.
PROC_BMCCPRS_STEP

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the BMC DASD MANAGER PLUS for DB2 utility. BMCCPRS references the utility that copies statistics from the DB2 catalog to the BMCSTATS tables.

This keyword is not included in the AJXPOFIN input stream.

PROC_BMCLOAD_NAME

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the BMC LOADPLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.

PROC_BMCLOAD_STEP

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the BMC LOADPLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.

PROC_BMCRECOVER_NAME

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the NGT Recover utility.

This keyword is not included in the AJXPOFIN input stream.

PROC_BMCRECOVER_STEP

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the NGT Recover utility.

This keyword is not included in the AJXPOFIN input stream.

PROC_BMCREORG_NAME

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the BMC REORG PLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.
PROC_BMCREORG_STEP

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the BMC REORG PLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.

PROC_BMCSTATS_NAME

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the BMCSTATS component of the BMC DASD MANAGER PLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.

PROC_BMCSTATS_STEP

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the BMCSTATS component of the BMC DASD MANAGER PLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.

PROC_BMCSTOP_NAME

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the BMC DASD MANAGER PLUS for DB2 utility.

BMCSTOP refers to the part of the utility that issues a DB2 STOP command on an object and verifies the completion of the command.

This keyword is not included in the AJXPOFIN input stream.

PROC_BMCSTOP_STEP

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the BMC DASD MANAGER PLUS for DB2 utility.

BMCSTOP refers to the part of the utility that issues a DB2 STOP command on an object and verifies the completion of the command.

This keyword is not included in the AJXPOFIN input stream.

PROC_BMCTRIG_NAME

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-
worklist JCL for the BMCTRIG component of the BMC DASD MANAGER PLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_BMCTRIG_STEP**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the BMCTRIG component of the BMC DASD MANAGER PLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_BMCUNLOAD_NAME**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the BMC UNLOAD PLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_BMCUNLOAD_STEP**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the BMC UNLOAD PLUS for DB2 utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_BMCUPRS_NAME**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the BMC DASD MANAGER PLUS for DB2 utility. BMCUPRS refers to the part of the utility that takes the statistics from the tables in the DASD MANAGER PLUS database (with the BMCSTATS utility) and updates the DB2 catalog with the statistics.

This keyword is not included in the AJXPOFIN input stream.

**PROC_BMCUPRS_STEP**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the BMC DASD MANAGER PLUS for DB2 utility. BMCUPRS refers to the part of the utility that takes the statistics from the tables in the DASD MANAGER PLUS database (with the BMCSTATS utility) and updates the DB2 catalog with the statistics.
This keyword is not included in the AJXPOFIN input stream.

**PROC_DSNUTILB_NAME**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the IBM DSNUTILB utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_DSNUTILB_STEP**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the IBM DSNUTILB utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_DSN1COPY_NAME**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the IBM DSN1COPY utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_DSN1COPY_STEP**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the IBM DSN1COPY utility.

This keyword is not included in the AJXPOFIN input stream.

**PROC_GEN_SET_VAR=N**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies whether to include an SLIB that generates SET statements in the JCL for variables that you can use in catalog procedures (PROCs).

This keyword is not included in the AJXPOFIN input stream.

**PROC_IDCAMS_NAME**

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the IBM IDCAMS program.

This keyword is not included in the AJXPOFIN input stream.
PROC_IDCAMs_STEP

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the IBM IDCAMS program.

This keyword is not included in the AJXPOFIN input stream.

PROC_IEFBR14_NAME

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for the IBM IEFBR14 job.

This keyword is not included in the AJXPOFIN input stream.

PROC_IEFBR14_STEP

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for the IBM IEFBR14 job.

This keyword is not included in the AJXPOFIN input stream.

PROC_TSO_NAME

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for IBM TSO.

This keyword is not included in the AJXPOFIN input stream.

PROC_TSO_STEP

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for IBM TSO.

This keyword is not included in the AJXPOFIN input stream.

PROC_USE=N

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies whether to generate a cataloged procedure (PROC) name instead of the EXEC PGM= statement for non-worklist JCL.

This keyword is not included in the AJXPOFIN input stream.
PROC_USER_DEF_STEP

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the EXEC job step in the cataloged procedure for non-worklist JCL for a user-defined program.

This keyword is not included in the AJXPOFIN input stream.

PROC_USER_DEFINED

For CATALOG MANAGER and DASD MANAGER PLUS, this keyword specifies the name of the cataloged procedure (PROC statement) for non-worklist JCL for a user-defined program.

This keyword is not included in the AJXPOFIN input stream.

PUNCH_DATACLASS

This keyword specifies the SMS definition for the data class associated with the punch data set.

PUNCH_EXPDT

This keyword specifies the expiration date of the punch data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyddd or yyyy/ddd.

PUNCH_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the punch data set.

PUNCH_PREFIX=&PREFIX..&WKID..&STEPN

When a limit key is changed, this keyword specifies the prefix of the data set that contains discard rows from the last partition of a table space after the table space is reorganized.

PUNCH_PRIQTY=1

This keyword specifies the primary allocation (in cylinders) for the punch data set if DATASETSIZING=N or if a sizing error occurs. Valid values are 1 through 99999.

PUNCH_RETPD

This keyword specifies the retention period for the punch data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.
**PUNCH_SECQTY=1**

This keyword specifies the secondary allocation (in cylinders) for the punch data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

**PUNCH_STORCLASS**

This keyword specifies the SMS definition for the storage class associated with the punch data set.

**PUNCH_UNIT=SYSDA**

This keyword specifies the unit name for the punch data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

**RCPY1_DATACLASS**

This keyword specifies the SMS definition for the data class associated with the recovery primary copy data set.

**RCPY1_DATACLASS_ALT**

This keyword specifies the SMS definition for the data class associated with the recovery primary copy data set (used if the threshold is exceeded).

**RCPY1_EXPDT**

This keyword specifies the expiration date of the recovery primary copy data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyddd or yyyy/ddd.

**RCPY1_MGMTCLASS**

This keyword specifies the SMS definition for the storage class associated with the recovery primary copy data set.

**RCPY1_MGMTCLASS_ALT**

This keyword specifies the SMS definition for the storage class associated with the recovery primary copy data set (used if the threshold is exceeded).

**RCPY1_PREFIX=&PREFIX..&OBNOD..P&PART**

This keyword specifies the prefix for the name of the recovery primary copy data set.
RCPY1_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the recovery primary copy data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

RCPY1_RETPD

This keyword specifies the retention period for the recovery primary copy data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

RCPY1_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the recovery primary copy data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

RCPY1_STACK=N

This keyword specifies whether to stack the recovery primary copy data set on a tape with data sets of the same type (Y or N).

RCPY1_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the recovery primary copy data set.

RCPY1_STORCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the recovery primary copy data set (used if the threshold is exceeded).

RCPY1_SUPPRESS_SUFF=N

This keyword specifies whether to suppress adding the DD name to the end of the name of the recovery primary copy data set (Y or N). If you specify Y, you must ensure that the data set name is unique.

RCPY1_THRESH=0

This keyword specifies the maximum anticipated size for the recovery primary copy data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.
RCPY1_UNIT=SYSDA

This keyword specifies the unit for the recovery primary copy data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

RCPY1_UNIT_ALT

This keyword specifies the alternate unit name for the recovery primary copy data set (used if the threshold value is exceeded). The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

RCPY2_DATACLASS

This keyword specifies the SMS definition for the data class associated with the recovery backup copy data set.

RCPY2_DATACLASS_ALT

This keyword specifies the SMS definition for the data class associated with the recovery backup copy data set (used if the threshold is exceeded).

RCPY2_EXPDT

This keyword specifies the expiration date of the recovery backup copy data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyddd or yyyy/ddd.

RCPY2_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the recovery backup copy data set.

RCPY2_MGMTCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the recovery backup copy data set (used if the threshold is exceeded).

RCPY2_PREFIX=&PREFIX..&OBNOD..P&PART

This keyword specifies the prefix for the name of the recovery backup copy data set.

RCPY2_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the recovery backup copy data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.
RCPY2_RETPD

This keyword specifies the retention period for the recovery backup copy data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

RCPY2_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the recovery backup copy data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 9999.

RCPY2_STACK=N

This keyword specifies whether to stack the recovery backup copy data set on a tape with data sets of the same type (Y or N).

RCPY2_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the recovery backup copy data set.

RCPY2_STORCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the recovery backup copy data set (used if the threshold is exceeded).

RCPY2_SUPPRESS_SUFF=N

This keyword specifies whether to suppress adding the DD name to the end of the name of the recovery backup copy data set (Y or N). If you specify Y, you must ensure that the data set name is unique.

RCPY2_THRESH=0

This keyword specifies the maximum anticipated size for the recovery backup copy data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

RCPY2_UNIT=SYSDA

This keyword specifies the unit for the recovery backup copy data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

RCPY2_UNIT_ALT

This keyword specifies the alternate unit name for the recovery backup copy data set (used if the threshold value is exceeded). The unit name must be
defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

**REBINDFAIL=N**

This keyword specifies whether worklist execution continues if a rebind fails. If REBINDFAIL=Y, worklist execution stops with a return code of 8. The stop is noted in the sync tables, and an Execution restart continues with the command that caused the failure. If REBINDFAIL=N, worklist execution continues.

**REBINDRC**

This keyword allows worklist execution to continue if a rebind fails, but returns a value for a final condition code instead of 4, the default value. Execution writes warning messages to AEXPRINT but does not post entries in the sync tables.

**RECOVERDOPT**

This keyword specifies the name of the installation options module for the NGT Recover utility. The BMC_RECOVER_OPTS AJXPOFIN keyword replaces this keyword. If both BMC_RECOVER_OPTS and RECOVERDOPT are included in the POF, the components use the value that is specified for RECOVERDOPT.

**REGION=0M**

This keyword defines the REGION parameter in the EXEC statement.

**REORG_MAPDB**

This keyword specifies the mapping database that ALTER and CHANGE MANAGER provide to the IBM REORG utility as an override to the value of the DB2 subsystem parameter REORG_MAPPING_DATABASE. CATALOG MANAGER and DASD MANAGER PLUS do not use this value. The name can be from 1 to 8 characters long, and cannot include symbolic variables.

*Note*

REORG_MAPDB applies to the dynamically and non-dynamically allocated data sets that the IBM REORG utility uses.

REORG_MAPDB also applies to the dynamically allocated data sets that the BMC REORG PLUS utility uses.

**REORG_MAPTAB**

This keyword specifies the name of the mapping table that the IBM REORG or BMC REORG PLUS utility uses to map the row IDs (RIDs) in the source table to the RIDs in the target table. ALTER and CHANGE MANAGER use
this value when reorganizing a table space by using an online reorganization (SHRLEVEL CHANGE). CATALOG MANAGER and DASD MANAGER PLUS do not use this value. The name can be from 1 to 72 characters long, and can contain the &ZUSER or &USERID symbolic variable.

**Note**
The REORG PLUS utility invokes the IBM DSNUTILB utility control program to enable certain features. If you have specified to use the REORG PLUS utility, you still need to specify mapping table information. For information about the features for which REORG PLUS invokes DSNUTILB, see the *REORG PLUS for DB2 Reference Manual*.

### REORGDOPT

This keyword specifies the name of the installation options module for the BMC REORG PLUS utility. The BMC_REORG_OPTS AJXPOFIN keyword replaces this keyword. If both BMC_REORG_OPTS and REORGDOPT are included in the POF, the components use the value that is specified for REORGDOPT.

### REPT_DATACLASS

This keyword specifies the SMS definition for the data class associated with the report data set.

### REPT_DATACLASS_ALT

This keyword specifies the SMS definition for the data class associated with the report data set (used if the threshold is exceeded).

### REPT_EXPDT

This keyword specifies the expiration date of the report data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyddd or yyyy/ddd.

### REPT_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the report data set.

### REPT_MGMTCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the report data set (used if the threshold is exceeded).

### REPT_PREFIX=&PREFIX..&WKID

This keyword specifies the prefix for the name of the report data set.
REPT_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the report data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

REPT RETPD

This keyword specifies the retention period for the report data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

REPT_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the report data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

REPT_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the report data set.

REPT_STORCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the report data set (used if the threshold is exceeded).

REPT_THRESH=0

This keyword specifies the maximum anticipated size for the report data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

REPT_UNIT=SYSDA

This keyword specifies the unit for the report data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

REPT_UNIT_ALT

This keyword specifies the alternate unit name for the report data set, if the threshold value is exceeded. The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.
RUNTIME_HLQ

This keyword specifies a high-level qualifier (HLQ) for ISPF data sets for the batch ISPF environment. During installation, if you chose to use the runtime enablement feature, the Installation System set this value to an HLQ for user runtime libraries. If you chose not to use the feature, the Installation System set the value to an HLQ for Execution.

SCHED_TRIG_CNTM_JOBS=N

This keyword specifies whether JCL Generation should generate BMC Control-M job schedule entries for jobs that do not contain IEFBR14 steps (Y or N). If the value is Y, JCL Generation generates an input data set (CNTLMSCH) and an output data set (CNTLMOUT) in the JCL for the BMCTRIG utility.

SORTWK_NBR=4

This keyword specifies the number of SORTWORK data sets. Valid values are 1 through 32.

SORTWK_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the DATAWORK, LOGSORT, or SORTWORK data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

SORTWK_SECQTY=2

This keyword specifies the secondary quantity (in cylinders) for the DATAWORK, LOGSORT, or SORTWORK data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

SORTWK_UNIT=SYSDA

This keyword specifies the name of the SORTWORK data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

SQLEXP_LOAD

This keyword specifies the name of the LINK library for the BMC SQL Explorer for DB2 product.

Tip

To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.
SRTOUT_DATACLASS

This keyword specifies the SMS definition for the data class associated with the SORTOUT data set.

SRTOUT_DATACLASS_ALT=CART

This keyword specifies the SMS definition for the data class associated with the SORTOUT data set (used if the threshold is exceeded).

SRTOUT_EXPDT

This keyword specifies the expiration date of the SORTOUT data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyddd or yyyy/ddd.

SRTOUT_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the SORTOUT data set.

SRTOUT_MGMTCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the SORTOUT data set (used if the threshold is exceeded).

SRTOUT_PREFIX=&PREFIX..&WKID..&STEPN

This keyword specifies the prefix for the name of the SORTOUT data set.

SRTOUT_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the SORTOUT data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

SRTOUT_RETPD=

This keyword specifies the retention period for the SORTOUT data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

SRTOUT_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the SORTOUT data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.
SRTOUT_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the SORTOUT data set.

SRTOUT_STORCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the SORTOUT data set (used if the threshold is exceeded).

SRTOUT_THRESH=0

This keyword specifies the maximum anticipated size for the SORTOUT data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero (0) indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

SRTOUT_UNIT=SYSDA

This keyword specifies the unit for the SORTOUT data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

SRTOUT_UNIT_ALT=CART

This keyword specifies the alternate unit name for the SORTOUT data set, if the threshold value is exceeded. The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

STEP_INCLUDE_MEMBER

This keyword specifies the name of a JCL member to be included after each step in the JCL.

STOPWAIT=3

This keyword specifies the number of intervals to wait for a DB2 STOP command to stop a database or table space. Zero (0) indicates that if the object does not stop, the worklist stops without waiting. The maximum value allowed for this keyword is 10 (which is 280 seconds).

STOPWTSECS=10

This keyword specifies the number of seconds to wait (during the first interval of the STOPWAIT keyword) for a DB2 STOP command to stop a database or table space.
SUPPRESS_COMMENTS=N

This keyword specifies whether to suppress the comments in the generated JCL (Y or N).

SYNCDELETE=N

This keyword specifies whether Execution should remove all sync entries when an Execution job completes with no errors (Y or N).

SYSEXC

This keyword specifies the name of the partitioned data set in which a REXX EXEC is a member.

**Tip**
To indicate the data set name for a different SSID, append the &SSID or &MSSID symbolic variable to the name.

SYSTEM_MLIB

This keyword specifies the name of the system ISPF message library. You specify the value for this keyword during installation.

SYSUT_DATACLASS

This keyword specifies the SMS definition for the data class associated with the SYSUT data set.

SYSUT_DATACLASS_ALT

This keyword specifies the SMS definition for the data class associated with the SYSUT data set (used if the threshold is exceeded).

SYSUT_EXPDT

This keyword specifies the expiration date of the SYSUT data set on tape. A data set cannot have an expiration date and a retention period. The valid values are **yyddd** or **yyyy/ddd**.

SYSUT_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the SYSUT data set.

SYSUT_MGMTCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the SYSUT data set (used if the threshold is exceeded).
SYSUT_PREFIX=&PREFIX..&WKID..&STEPN

This keyword specifies the prefix for the name of the SYSUT and WORKDDN data sets.

SYSUT_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the SYSUT data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

SYSUT_RETPD

This keyword specifies the retention period for the SYSUT data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

SYSUT_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the SYSUT data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

SYSUT_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the SYSUT data set.

SYSUT_STORCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the SYSUT data set (used if the threshold is exceeded).

SYSUT_THRESH=0

This keyword specifies the maximum anticipated size for the SYSUT data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

SYSUT_UNIT=SYSDA

This keyword specifies the unit for the SYSUT data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

SYSUT_UNIT_ALT

This keyword specifies the alternate unit name for the SYSUT data set (used if the threshold value is exceeded). The unit name must be defined in the
TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

SZDEVT=3390

This keyword specifies the device type used in data set sizing. Valid values are 3380 or 3390.

TAPE_EXPDT

This keyword specifies the expiration date for the tape set. The value specified for this keyword is valid for all tape data sets for which an expiration date is not specified.

TAPE_RETPD

This keyword specifies the retention period for the tape set. The value specified for this keyword is valid for all tape data sets for which a retention period is not specified.

TAPE_VOLCNT

This keyword specifies the maximum number of tape volumes. Valid values are 0 through 255.

TAPE1=CART
TAPE2=TAPE
TAPE3=TAPE

These keywords define the names of the tape units for an installation.

TEMPLATE_DSN

For CATALOG MANAGER, this keyword specifies the name of the data set that contains member names for TEMPLATE utility control statements.

TEMPUNIT=SYSDA

This keyword defines the name of the unit that the components use for temporary files. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

TIMEPARM

This keyword indicates the time limit in minutes for each step in a batch job stream.
TRTCH

This keyword specifies the parity, data conversion, translation, and compression for 7-track drives. The following values are valid:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>Does not use a 7-track drive</td>
</tr>
<tr>
<td>C</td>
<td>Uses odd parity, conversion on, and translation off</td>
</tr>
<tr>
<td>E</td>
<td>Uses even parity, conversion off, and translation off</td>
</tr>
<tr>
<td>T</td>
<td>Uses odd parity, conversion off, and translation on</td>
</tr>
<tr>
<td>ET</td>
<td>Uses even parity, conversion off, and translation on</td>
</tr>
<tr>
<td>COMP</td>
<td>Uses data compression on</td>
</tr>
<tr>
<td>NOCOMP</td>
<td>Uses data compression off</td>
</tr>
</tbody>
</table>

TSOPROGRAM

This keyword specifies an alternate TSO monitor program for standard JCL. TSOPROGRAM is available for nonworklist JCL.

TSOSUBEXIT=N

This keyword specifies whether to use a TSO submit exit to generate job cards. If TSOSUBEXIT=Y, no job cards are put in the JCL (Y or N).

ULLQ

This keyword specifies the LLQ for user-defined data sets for the installation environment. During installation, if you chose to use the runtime enablement feature, the Installation System set this value to blank, and you should not change the value. If you chose not to use the feature, the Installation System set the value to UDB.

**Note**

Changing the qualifier of the data sets might cause unpredictable results. Do not change the LLQ for the data sets.

UNLD_FREF_DATACLASS

For ALTER and CHANGE MANAGER, this keyword specifies the SMS definition for the data class associated with the file reference (SYSREC) data set.

UNLD_FREF_DIRBLOCK = 250

For ALTER and CHANGE MANAGER, this keyword specifies the number of directory blocks for the file reference (SYSREC) data set.
UNLD_FREF_MGMTCLASS

For ALTER and CHANGE MANAGER, this keyword specifies the SMS
definition for the storage class associated with the file reference (SYSREC)
data set.

UNLD_FREF_PREFIX = &PREFIX..&MSSID..&WORKID8

For ALTER and CHANGE MANAGER, this keyword specifies the prefix for
the name of the file reference (SYSREC) data set.

UNLD_FREF_PRIQTY = 10

For ALTER and CHANGE MANAGER, this keyword specifies the primary
allocation (in cylinders) for the file reference (SYSREC) data set. Valid values
are 1 through 99999.

UNLD_FREF_SECQTY = 2

For ALTER and CHANGE MANAGER, this keyword specifies the secondary
allocation (in cylinders) for the file reference (SYSREC) data set. Valid values
are 1 through 99999.

UNLD_FREF_STORCLASS

For ALTER and CHANGE MANAGER, this keyword specifies the SMS
definition for the storage class associated with the file reference (SYSREC)
data set.

UNLD_FREF_SUPPR_SUFF=N

For ALTER and CHANGE MANAGER, this keyword specifies whether to
suppress adding the DD name to the end of the name of the file reference
(SYSREC) data set (Y or N). If you specify Y, you must ensure that the data
set name is unique.

UNLD_FREF_UNIT = SYSDA

For ALTER and CHANGE MANAGER, this keyword specifies the unit for
the file reference (SYSREC) data set. The value of the unit can be a name from
1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the
JCL, specify NONE.

UNLD1_DATACLASS

This keyword specifies the SMS definition for the data class associated with
the primary unload (SYSREC) data set.
UNLD1_DATACLASS_ALT

This keyword specifies the SMS definition for the data class associated with the primary unload (SYSREC) data set (used if the threshold is exceeded).

UNLD1_EXPDT

This keyword specifies the expiration date of the primary unload (SYSREC) data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyddd or yyyy/ddd.

UNLD1_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the primary unload (SYSREC) data set.

UNLD1_MGMTCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the primary unload (SYSREC) data set (used if the threshold is exceeded).

UNLD1_PREFIX=&USERID..&MSSID..&WORKID8

This keyword specifies the prefix for the name of the primary unload (SYSREC) data set.

UNLD1_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the primary unload (SYSREC) data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

UNLD1_RETPD

This keyword specifies the retention period for the primary unload (SYSREC) data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

UNLD1_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the primary unload (SYSREC) data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

UNLD1_STACK=N

This keyword specifies whether to stack the primary unload (SYSREC) data set on a tape with data sets of the same type (Y or N).
UNLD1_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the primary unload (SYSREC) data set.

UNLD1_STORCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the primary unload (SYSREC) data set (used if the threshold is exceeded).

UNLD1_SUPPRESS_SUFF=N

This keyword specifies whether to suppress adding the DD name to the end of the name of the primary unload (SYSREC) data set (Y or N). If you specify Y, you must ensure that the data set name is unique.

UNLD1_THRESH=0

This keyword specifies the maximum anticipated size for the primary unload (SYSREC) data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

UNLD1_UNIT=SYSDA

This keyword specifies the unit for the primary unload (SYSREC) data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

UNLD1_UNIT_ALT

This keyword specifies the alternate unit name for the primary unload (SYSREC) data set (used if the threshold value is exceeded). The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

UNLD2_DATACLASS

This keyword specifies the SMS definition for the data class associated with the backup unload (SYSREC) data set.

UNLD2_DATACLASS_ALT

This keyword specifies the SMS definition for the data class associated with the backup unload (SYSREC) data set (used if the threshold is exceeded).
UNLD2_EXPDT

This keyword specifies the expiration date of the backup unload (SYSREC) data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyddd or yyyy/ddd.

UNLD2_MGMTCLASS

This keyword specifies the SMS definition for the storage class associated with the backup unload (SYSREC) data set.

UNLD2_MGMTCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the backup unload (SYSREC) data set (used if the threshold is exceeded).

UNLD2_PREFIX=&USERID..&MSSID..&WORKID8

This keyword specifies the prefix for the name of the backup unload (SYSREC) data set.

UNLD2_PRIQTY=10

This keyword specifies the primary allocation (in cylinders) for the backup unload (SYSREC) data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

UNLD2_RETPD

This keyword specifies the retention period for the backup unload (SYSREC) data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

UNLD2_SECQTY=2

This keyword specifies the secondary allocation (in cylinders) for the backup unload (SYSREC) data set if DATASETSIZING=N or if an error in sizing occurs. Valid values are 1 through 99999.

UNLD2_STACK=N

This keyword specifies whether to stack the backup unload (SYSREC) data set on a tape with data sets of the same type (Y or N).

UNLD2_STORCLASS

This keyword specifies the SMS definition for the storage class associated with the backup unload (SYSREC) data set.
UNLD2_STORCLASS_ALT

This keyword specifies the SMS definition for the storage class associated with the backup unload (SYSREC) data set (used if the threshold is exceeded).

UNLD2_SUPPRESS_SUFF=N

This keyword specifies whether to suppress adding the DD name to the end of the name of the backup unload (SYSREC) data set (Y or N). If you specify Y, you must ensure that the data set name is unique.

UNLD2_THRESH=0

This keyword specifies the maximum anticipated size for the backup unload (SYSREC) data set, in cylinders. If this value is exceeded, the alternate unit and the alternate SMS keywords will be used. Zero indicates that a threshold is not specified for the unit. If zero is specified, an alternate unit and the alternate SMS keywords will not be used.

UNLD2_UNIT(SYSDA)

This keyword specifies the unit for the backup unload (SYSREC) data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

UNLD2_UNIT_ALT

This keyword specifies the alternate unit name for the backup unload (SYSREC) data set (used if the threshold value is exceeded). The unit name must be defined in the TAPE1, TAPE2, or TAPE3 keywords. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

UNLD3_DATACLASS

For the BMC UNLOAD PLUS utility in the Database Administration solution, this keyword specifies the SMS definition for the data class associated with the ROWID SYSREC data set.

UNLD3_EXPDT

For the BMC UNLOAD PLUS utility in the Database Administration solution, this keyword specifies the expiration date of the ROWID SYSREC data set on tape. A data set cannot have an expiration date and a retention period. The valid values are yyyddd or yyyy/ddd.
UNLD3_MGMTCLASS

For the BMC UNLOAD PLUS utility in the Database Administration solution, this keyword specifies the SMS definition for the storage class associated with the ROWID SYSREC data set.

UNLD3_PREFIX=&USERID..&MSSID..&WORKID8

For the BMC UNLOAD PLUS utility in the Database Administration solution, this keyword specifies the prefix for the name of the ROWID SYSREC data set.

UNLD3_RETPD

For the BMC UNLOAD PLUS utility in the Database Administration solution, this keyword specifies the retention period for the ROWID SYSREC data set on tape. A data set cannot have an expiration date and a retention period. Valid values are 1 through 9999.

UNLD3_STORCLASS

For the BMC UNLOAD PLUS utility in the Database Administration solution, this keyword specifies the SMS definition for the storage class associated with the ROWID SYSREC data set.

UNLD3_SUPPRESS_SUFF=N

For the BMC UNLOAD PLUS utility in the Database Administration solution, this keyword specifies whether to suppress adding the DD name to the end of the name of the ROWID SYSREC data set (Y or N). If you specify Y, you must ensure that the data set name is unique.

UNLD3_UNIT=SYSDA

For the BMC UNLOAD PLUS utility in the Database Administration solution, this keyword specifies the unit for the ROWID SYSREC data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

UNLD4_DATACLASS

For version 1.2 or earlier of the LOB DATA MOVER program in the Database Administration or BMC Object Administration for DB2 solution, this keyword specifies the SMS definition for the data class associated with the large object (LOB) SYSREC data set.

UNLD4_MGMTCLASS

For version 1.2 or earlier of the LOB DATA MOVER program in the Database Administration or BMC Object Administration for DB2 solution, this
keyword specifies the SMS definition for the storage class associated with the large object (LOB) SYSREC data set.

**UNLD4_PREFIX=&USERID..&MSSID..&WORKID8**

For version 1.2 or earlier of the LOB DATA MOVER program in the Database Administration or BMC Object Administration for DB2 solution, this keyword specifies the prefix for the name of the large object (LOB) SYSREC data set.

**UNLD4_STORCLASS**

For version 1.2 or earlier of the LOB DATA MOVER program in the Database Administration or BMC Object Administration for DB2 solution, this keyword specifies the SMS definition for the storage class associated with the large object (LOB) SYSREC data set.

**UNLD4_SUPPRESS_SUFF=N**

For version 1.2 or earlier of the LOB DATA MOVER program in the Database Administration or BMC Object Administration for DB2 solution, this keyword specifies whether to suppress adding the DD name to the end of the name of the large object (LOB) SYSREC data set (Y or N). If you specify Y, you must ensure that the data set name is unique.

**UNLD4_UNIT=SYSDA**

For version 1.2 or earlier of the LOB DATA MOVER program in the Database Administration or BMC Object Administration for DB2 solution, this keyword specifies the unit for the large object (LOB) SYSREC data set. The value of the unit can be a name from 1 to 8 characters long, blank, or NONE. To omit the UNIT parameter from the JCL, specify NONE.

**UNLOADDOPT**

This keyword specifies the name of the installation options module for the BMC UNLOAD PLUS utility. The BMC_UNLOAD_OPTS AJXPOFIN keyword replaces this keyword. If both BMC_UNLOAD_OPTS and UNLOADDOPT are included in the POF, the components use the value that is specified for UNLOADDOPT.

**USER_HLQ**

This keyword specifies the high-level qualifier (HLQ) used by user defined data sets. This HLQ is also used for the runtime enablement (RTE) data sets.

**USE_NGT_AUTO=N**

This keyword controls the behavior of Analysis when the product generates an -NGTU command in the worklist.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Uses the OUTPUT syntax.</td>
</tr>
<tr>
<td>Y</td>
<td>Omits the OUTPUT syntax.</td>
</tr>
</tbody>
</table>

**USER_VAR1_CHAR**
**USER_VAR2_CHAR**
**USER_VAR3_CHAR**
**USER_VAR4_CHAR**
**USER_VAR5_CHAR**

These keywords specify user-defined character variables. Each variable has a corresponding symbolic variable that you can use in job cards or data set prefixes. The maximum length of a variable name is eight characters.

**WORK_DATACLASS**

This keyword defines the Data Facility Storage Management Subsystem (DFSM$ or SMS) data class name that the components use at data-set allocation time to define allocation attributes of the data set. A data class name is not required, even for SMS data sets. This option appears as "DATACLAS=" in the JCL for workfiles.

**WORK_MGMTCLASS**

This keyword defines the DFSMS or SMS management class name that the components use at data-set allocation time to define the migration, retention, and backup requirements of the data set. This option appears as "MGMTCLAS=" in the JCL for workfiles.

**WORK_STORCLASS**

This keyword defines the DFSMS or SMS storage class name that the components use at data-set allocation time to define processing requirements of the data set. This option appears as "STORCLAS=" in the JCL for nontape work files.
Worklist commands

A worklist consists of a number of commands that the Execution component uses to set up and run DB2 SQL statements, DB2 commands and utilities, and BMC programs.

Some of these commands are not generated by the Analysis component, but all of the commands are listed here for complete information. If you need to use one of these commands, you can manually add it to a worklist.

File format

Worklist and CDL files consist of a sequence of commands that occupy single or multiple lines.

The file structure is sequential and fixed in length, with 80-character records. Table 19 on page 259 defines the record layout of the worklist file and the record layout of the CDL file.

Table 19: Worklist file format

<table>
<thead>
<tr>
<th>Column number</th>
<th>Layout</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Line identifier</td>
<td>A dash (-) that indicates the beginning of a command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>An asterisk (*) indicates a comment. If there is no indicator, then this line is a continuation of the previous Command line.</td>
</tr>
<tr>
<td>2-5</td>
<td>Command identifier</td>
<td>A four-character code that identifies the command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The command identifier appears only on the first line of the command.</td>
</tr>
<tr>
<td>6</td>
<td>Blank</td>
<td>A blank character (required)</td>
</tr>
<tr>
<td>Column number</td>
<td>Layout</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>7-12</td>
<td>Command sequence number</td>
<td>A six-digit number, right-justified and zero-filled, that indicates the command sequence. Each command has a unique number. These numbers must remain in order. You might add and delete commands, however, you must keep the commands in ascending sequence. Command sequence numbers are generated in increments of 50 to allow you to insert commands as needed. Although you can change a command sequence number, doing so results in a hash failure. The command sequence number appears only on the first line of a command.</td>
</tr>
<tr>
<td>13</td>
<td>Blank</td>
<td>A blank character (required)</td>
</tr>
<tr>
<td>14-72</td>
<td>Command text</td>
<td>Free-format text that includes the information that is needed to execute the command. For continuation lines, the text appears in columns 2 through 72. <strong>Note:</strong> No implicit break or space exists between column 72 of one line and column 2 of the following continuation line. Keywords cannot be split over multiple lines.</td>
</tr>
<tr>
<td>73-80</td>
<td>Hash verification number</td>
<td>A numerical hash value that indicates the contents of the command. Several components use the number to detect changed and inserted commands. Do not insert or modify this code. The hash number appears only on the last line of a command.</td>
</tr>
</tbody>
</table>

### Multi-line commands

Worklist files can consist of a sequence of commands that occupy multiple lines. Multi-line commands consist of the following identifiers and numbers:

- The line identifier, command identifier, command sequence number, and short command text on the first line.
- A blank in the line identifier column, with long command text on continuation lines.
- The hash verification number on the last line.
Comment lines

Comment lines consist of an asterisk (*) for the line identifier and comment text for the remainder of the line.

Comments do not use sequence numbers or hash verification numbers.

Note

You cannot embed comment lines within a command. A comment signals the end of the previous command.

ALTER and CHANGE MANAGER worklist commands

This section describes each command in detail and provides an example of the command.

-AMS (IDCAMS execution)

The -AMS command executes the IBM IDCAMS utility to define VSAM data sets.

More than one statement is supported per -AMS command. The IBM Storage Management Subsystem (SMS) parameters STORAGECLASS, MANAGEMENTCLASS, and DATACLASS are supported. Figure 3 on page 261 shows an example of the -AMS command.

Note

This command can be used for other IDCAMS functions as well.

Zero (0) is the highest return code from IDCAMS that the ALTER or CHANGE MANAGER products accept. To allow a higher return code, use the IDCAMS SET command to reset the code to 0. For information about the IDCAMS utility, see the IBM documentation.

Figure 3: -AMS command

```
-AMS  002800
  DEFINE CLUSTER
    (NAME(DBAPCAT.DSNDBC.X5DB0002.X5IXPTA1.10001.A003)
    LINEAR
    STORCLAS(AUSCLASS)
    DATACLAS(LINEAR)
    MGMTCLAS(DBSTNDRD)
```
-AUTH (Authorization switching)

The -AUTH command executes a BMC program that changes the execution authorization ID to the authorization ID that is specified in the text of the command.

This function allows the creator of an object to be different from the user who is submitting the worklist. Figure 4 on page 262 shows an example of the -AUTH command.

Figure 4: -AUTH command

```
-AUTH 000160 JEO
-SQL 000170
CREATE TABLESPACE PROPERTY
IN RESTATE
USING
STOGROUP DB2030
PRIQTY 3
SECQTY 3
ERASE NO
FREEPAGE 5
PCTFREE 25
BUFFERPOOL BP0
LOCKSIZE PAGE
CLOSE YES
```

Note the following items when the -AUTH command is used in a worklist:

- If the worklist uses a global authorization ID (GLID), an -AUTH command with a value of USER sets the current authorization ID to the GLID.

- If the worklist does not use a GLID, an -AUTH command with a value of USER sets the current authorization ID to the ID of the user who is executing the worklist.

- If Analysis cannot determine the qualifier of a table that is used in a view, the -AUTH command is specified with a value of UNKNOWN in the text field.

- This command can exit to an installation supplied security exit. You can use this exit for additional authorization ID control. For information about the security exit and about using 8-byte authorization IDs, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 1*.

- For information about the AUTHSW option, see “Authorizations used for authorization switching” on page 362. This option determines the commands that are used for authorization ID switching.
The -BASE command causes Execution to invoke the Baseline component of
CHANGE MANAGER.

Required parameters are the subsystem ID, work ID, baseline profile name, and
source type. Figure 5 on page 263 shows an example of the -BASE command.

Figure 5: -BASE command

```
BASE 000250 BLP profle RDAMAF.FULLREC
 UNLO ADE T YPE B
 BLNAME RDAMAF. FULLREC####
 WORKID RDAMAF. FULLREC
 SSID DB71
 SOURCE T YPE CATALOG
```

Optionally, you can include the BLNAME parameter, which passes a baseline name
to Baseline. If BLNAME is not specified, Baseline will use the baseline name
template that the specified baseline profile provides. If the baseline profile does not
contain a name template, however, BLNAME is required.

The CM/PILOT component of CHANGE MANAGER also uses the -BASE command
in a CM/PILOT worklist, as shown in Figure 6 on page 263.

Figure 6: -BASE command in a CM/PILOT worklist

```
BASE 000600
 SSID DBAH
 BLP profle RDAMAF.DDL
 BLNAME RDAMAF2.DDL@@@@
 SOURCE T YPE DDL
 DELETEAGE 0000
 RETAINMAX 0000
 UNLO ADE T YPE B
 WORKID RDAMAF2.RECSR
```
Analysis inserts the UNLOADTYPE \( x \) parameter when it creates a worklist that requests a full-recovery baseline. This parameter records the unload and load options that are specified. The following values are valid for \( x \):

- **B** indicates that the BASIC UNLOAD or the UNLOAD PLUS utility unloaded the data. The IBM LOAD utility must load the data.
  
The B option is not valid when you are unloading tables that contain XML or LOB columns. BASIC UNLOAD cannot unload these columns. In addition, you cannot use UNLOAD PLUS and IBM LOAD to unload and load these columns.

- **M** indicates that the BASIC UNLOAD or the UNLOAD PLUS utility unloaded the data. The LOADPLUS utility must load the data.
  
  If you are using the BASIC UNLOAD utility, the M option is not valid when you are unloading tables that contain XML or LOB columns. BASIC UNLOAD cannot unload these columns.

- **D** indicates that the IBM UNLOAD utility unloaded the data. The IBM LOAD utility must load the data.

  This information is stored with the baseline for use during a full recovery, and is displayed in the baseline report.

---

**-BASF (Mark baseline recoverable)**

The -BASF command marks baselines as recoverable after unloads are complete.

Analysis adds the command in a new section after the unload section and follows with a -SYNC command. The baseline process only marks the baseline as complete, not as recoverable. Figure 7 on page 264 shows an example of the -BASF command.

---

**Note**

This command only applies to worklists taking full recovery baselines.
-BEGG (Begin global commands)

For the Database Administration or BMC Object Administration for DB2 solution, the -BEGG command indicates the start of a CHANGE MANAGER worklist that can be processed in parallel.

The command must be the first command in the worklist, as shown in Figure 8 on page 265.

**Figure 8: -BEGG and -ENDG commands**

- **BEGG 000050**
- **TIME 000100 '2010-07-10-09.05.01.705633'**
- **SSID 000150 DEBA**
- **JCLP 000200 MIGR MSSID DEBA**
- **WKID 000250 ACM.MG0829G**
- **ENDG 000300**

The -BEGG command is a global worklist command; that is, it is sent to each initiator that is started to process units of work (UOW) from a parallel worklist. The -BEGG command must have a corresponding -ENDG command in the worklist. The statements between the -BEGG and -ENDG commands are sent to each initiator when a worklist is initially executed and when it is restarted. For more information about the -ENDG command, see “-ENDG (End global commands)” on page 309.

-BEGP (Begin parallel processing)

For the Database Administration or BMC Object Administration for DB2 solution, the -BEGP command indicates the start of UOWs in the worklist that are processed in parallel.

The -BEGP command, along with the -ENDP command, encloses one or more UOWs, as shown in Figure 9 on page 266. (UOWs are enclosed with -BEGU and -ENDU commands.) All of the UOWs within the -BEGP and -ENDP commands must complete processing before the next UOW outside of the -BEGP and -ENDP group can be processed.
The following worklist commands can be processed in parallel:

- **-BMCC**—NGT Copy utility
- **-BMCD**—UNLOAD PLUS utility
- **-BMCK**—CHECK PLUS utility
- **-BMCL**—LOADPLUS utility
- **-BMCS**—BMC BMCSTATS utility
- **-CHEK**—IBM CHECK DATA utility
- **-RNST**—IBM RUNSTATS utility

For more information about the **-ENDP** command, see “**-ENDP (End parallel processing)**” on page 310.

**Figure 9: -BEGP and -ENDP commands**

```
-BEGP 000550
-BEGU 000600
-BMCD 000650
  UNLOAD UNLOADDN (SYSRECO1)
  ACTIVE (YES,NO)
  DISCARDS 1
  NULLTYPE T1 NULLCHAR ?
  FORMAT BMCLOAD
  SELECT *
  FROM   ACMX06.T_X60SS
  ;
  OUTPUT SYSRECO1
  UNIT SYSDA
  DSNAME 'RDAMCG.MG0829G.ACMX06.T@X60SS.U0001'
-SYN D00700 UNLOAD OF TABLE ACMX06.T_X60SS COMPLETE
-ENDU 000750
-JCLP 000800 BMCD IPPARTS 4
-BEGU 000850
-BMCD 000900
  UNLOAD UNLOADDN (SYSR)
  ACTIVE (YES,NO)
  DISCARDS 1
  NULLTYPE T1 NULLCHAR ?
  FORMAT BMCLOAD
  SELECT *
  FROM   ACMX01.T_X01PS
  ORDER BY COLUMN_1 , COLUMN_2 , COLUMN_3 , COLUMN_4 , COLUMN_5 , COLUMN_7 , COLUMN_8 , COLUMN_9 , COLUMN_10 , COLUMN_11 , COLUMN_12 , COLUMN_13 , COLUMN_14 , COLUMN_15 , COLUMN_17 , COLUMN_18 , COLUMN_19 , COLUMN_20 , COLUMN_21
  ;
  OUTPUT SYSR
  UNIT SYSDA
  DSNAME 'RDAMCG.MG0829G.ACMX01.T@X01PS.U0002.P&PART'
-SYNC D00950 UNLOAD OF TABLE ACMX01.T_X01PS COMPLETE
-ENDU 001000
-JCLP 001050 BMCD IPPARTS 5
-BEGU 001100
```
-BEGU (Begin unit of work)

For the Database Administration or BMC Object Administration for DB2 solution, the -BEGU command indicates the start of a UOW.

The UOW can consist of a single command or several commands. The commands within the UOW are executed sequentially. The -ENDU command indicates the end of a UOW.

Special considerations apply to UOWs that contain -SQL and -AUTH commands. These UOWs are not executed in parallel, and they typically consist of an extensive number of lines. However, the worklist limits each UOW to 100,000 lines. Analysis divides this type of UOW into smaller portions by inserting -BEGU and -ENDU commands after every 90,000 lines of the worklist. For more information about the -ENDU command, see “-ENDU (End unit of work)” on page 310.

Figure 10 on page 267 shows an example of the -BEGU command.

Figure 10: -BEGU and -ENDU commands

-BEGU 004800
-BMCC 004850
OUTPUT DCPYLOCP
UNIT SYSDA
DSNAME &UID.&UTIL.&DB.&TS.LP
COPY TABLESPACE ACMX01S.X60SS
COPYDDN (DCPYLOCP)
-SYNC 004900 COPY FOR TS ACMX01S.X60SS COMPLETE
-ENDU 004950
-ENDP 004950

-BIND (Bind)

The -BIND command invokes the DB2 BIND command.
This command is valid under an authorization ID switch. The parameters that are used are those in the DB2 catalog. Figure 11 on page 268 shows an example of the -BIND command.

**Figure 11: -BIND command**

```
-AUTH 000910 PAYDB
-BIND 000920 BIND ACTION(REPLACE) +
       RETAIN ISOLATION(CS) +
       VALIDATE(BIND) +
       PLAN(PAYROLL) +
       OWNER(ALUD41) +
       MEMBER(PAY1,PAY2)
```

*Note*

The TSO continuation character (+) is required for multiple line entries.

**-BMCB (BMC BASIC UNLOAD utility)**

The -BMCB command invokes the BMC BASIC UNLOAD utility to unload data for the tables that need to be rebuilt.

The BASIC UNLOAD utility is included with ALTER and CHANGE MANAGER. The utility runs as a subtask.

ALTER and CHANGE MANAGER run the BASIC UNLOAD utility to perform the following functions:

- Unload data by using a DB2 SQL SELECT statement
- Migrate tables with ROWID columns

The BASIC UNLOAD utility uses a portion of the functionality of the UNLOAD PLUS utility. For more information about the UNLOAD PLUS utility, see the *UNLOAD PLUS for DB2 Reference Manual*.

**Use of -BMCB to unload data**

The BASIC UNLOAD utility uses DB2 dynamic SQL programming techniques to process a SELECT statement and read data in a table.

The utility retains the format of the data (EBCDIC, ASCII, or Unicode). The BASIC UNLOAD utility takes advantage of the functionality that the DB2 SQL SELECT
statement provides. Figure 12 on page 269 shows an example of the -BMCB command.

Figure 12: -BMCB command

```
-BMCB 000250
  UNLOAD UNLOADDN (SYSR1001)
  ACTIVE (YES,NO)
  DISCARDS 1
  NULLTYPE T1 NULLCHAR ?
  EBCDIC CCSID(37,0,0)
  DIRECT NO
  SELECT *
  FROM   RDACRJ.NEWCLAIM
```

### Use of -BMCB to migrate tables with ROWID columns

You can modify or migrate tables that contain ROWID columns with data.

If the ROWID column is defined as GENERATED ALWAYS, ALTER and CHANGE MANAGER omit the ROWID column in the UNLOAD and LOAD statements. If the ROWID column is defined as GENERATED BY DEFAULT, ALTER and CHANGE MANAGER include the ROWID column in the UNLOAD and LOAD statements. The column is unloaded last in the sequence of columns and is also loaded last, even though it might not be the last column in the table. When a table that contains a ROWID column is unloaded, Analysis lists all of the columns that are unloaded in the UNLOAD statement in the worklist.

### -BMCC (NGT Copy utility)

The -BMCC command invokes the NGT Copy utility.

The -BMCC command can be processed in parallel. For more information about the worklist parallelism feature of the Database Administration and BMC Object Administration for DB2 solutions, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

You can run the NGT Copy utility to perform the following functions:

- Dynamically allocate copy data sets
- Create partition-level image copies
- Create a sequential migration file
Use of -BMCC to dynamically allocate copy data sets

The NGT Copy utility can dynamically allocate the data sets that are used to create valid image copies.

JCL Generation uses the symbolic variable that is specified in the JCL Generation Data Set Options for Copies panels to create an OUTPUT descriptor for Analysis. When generating the -BMCC command, Analysis specifies the OUTPUT descriptor for local and recovery primary and backup copies. The DSNAME parameter in the descriptor specifies the data set name for the object that is copied. Figure 13 on page 270 shows an example of a -BMCC command that uses dynamic allocation.

Figure 13: -BMCC command—dynamic allocation of data sets

```
-BMCC 004800
  OUTPUT DCPYLOCP
  UNIT SYSDA
  DSNAME RDAMCG.&UTIL.A.&DB.&TS.P&PART
  COPY TABLESPACE ACMX06A.X61NS
  COPYDDN (DCPYLOCP)
```

The P&PART symbolic variable is used in the DSNAME parameter to ensure that the data set name for each object is unique for partition-level image copies.

If you want to stack the data sets on a tape with data sets of the same type, the product groups the OUTPUT descriptors and COPY statements in one -BMCC command, as shown in Figure 14 on page 270.

Note

The data sets for table spaces that are implicitly created cannot be stacked on tape.

Figure 14: -BMCC command—dynamic allocation with tape stacking

```
-BMCC 002500
  OUTPUT DCPYLOCP
  UNIT CART
  DSNAME ACM.&UTIL.A.&DB.&TS
  VOLCNT 99
  EXPDT 2002/200
  STACK YES
  OUTPUT DCPYREMP
    UNIT CART
    DSNAME ACM.&UTIL.C.&DB.&TS
    VOLCNT 99
    EXPDT 2002/200
    STACK YES
    COPY TABLESPACE ACMX01L.X05SS
      COPYDDN (DCPYLOCP)
      RECOVERYDDN (DCPYREMP)
    TABLESPACE ACMX01L.X04NS
      COPYDDN (DCPYLOCP)
      RECOVERYDDN (DCPYREMP)
```
Use of `-BMCC` to create partition-level image copies

Analysis can generate a partition-level image copy of a partitioned table space or index in a database when you select to use NGT Copy as the copy utility.

Figure 15 on page 271 shows an example of the `-BMCC` command that is used when partition-level image copies are requested for a partitioned index. The ALL option in the OPTIONS parameter indicates that the entire partitioned index is copied.

**Figure 15: `-BMCC` command—partition-level image copies**

```
-SPAC 001100 DDNAME C002 BMCC IX C7PCB1C.IX37A1
-BMCC 001150
  OPTIONS IXDSNUM ALL
  COPY INDEX C7PCB1C.IX37A1
  COPYDDN (C0020001)
  DSNUM 1
  INDEX C7PCB1C.IX37A1
  COPYDDN (C0020002)
  DSNUM 2
  INDEX C7PCB1C.IX37A1
  COPYDDN (C0020003)
  DSNUM 3
```

Use of `-BMCC` to create sequential migration files

The NGT Copy EXPORT command and NGT Recover IMPORT command enable you to migrate an image copy or set of image copies within a DB2 subsystem or to another DB2 subsystem.

The EXPORT and IMPORT commands comprise the Copy Migration feature of the Recovery Management, BMC Recovery for DB2, BMC Object Administration for DB2, BMC Next Generation Technology Database Administration for DB2, and Database Administration solutions.

NGT Copy invokes the NGT Copy EXPORT command on the sending subsystem during phase 1 of the migration to collect information on one or more table spaces. This information includes object identifiers, the names of image copies, and other information relating to the sending table spaces. The EXPORT command includes this information in a sequential migration file.

Figure 16 on page 271 shows an example of the `-BMCC` command that is used with the EXPORT command.

**Figure 16: `-BMCC` command—EXPORT command**

```
-BMCC 000700
  OPTIONS MAXTASKS (1,1)
  OUTPUT EXPFILE
  UNIT SYSALLDA
  EXPOUT YES
  DSNNAME RDADQL.DEGA.WORKIDXX.EX000001
  EXPORT TABLESPACE DQC07090.TS07090A, DQ2CLG2.T03PBGY
```
The EXPORT command uses the options listed in Figure 16 on page 271.

### Table 20: EXPORT command options

<table>
<thead>
<tr>
<th>Option</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTIONS MAXTASKS</td>
<td>Instructs NGT Copy to use only one task, so that NGT Copy creates only one EXPORT file and includes all of the table spaces in the file</td>
</tr>
<tr>
<td>EXPORT TABLESPACE</td>
<td>Provides an explicit list of table spaces to include in the scope of the migration. To include history tables in the EXPORT file, you must include the parent table space in the NGT Copy scope of the migration.</td>
</tr>
<tr>
<td>ATLOGPOINT LASTFULLCOPY</td>
<td>Instructs NGT Copy to use the last full image copy</td>
</tr>
<tr>
<td>AUX ALL</td>
<td>Instructs NGT Copy to include auxiliary objects in the image copy. You do not have to explicitly specify the auxiliary objects.</td>
</tr>
<tr>
<td>INDEXES YES</td>
<td>Instructs NGT Copy to include indexes in the image copy</td>
</tr>
<tr>
<td>REUSE YES</td>
<td>Allocates the EXPORT file (if the file does not already exist), replaces the contents of the EXPORT file (if the file does exist), and reprocesses all table spaces when restarting or starting over a worklist. You must use REUSE YES if you are restarting a worklist.</td>
</tr>
</tbody>
</table>

**Notes for -BMCC**

For some objects, Analysis generates the -BMCC command differently.

Note the following items when the -BMCC command is used in a worklist:
■ The utility command starts on the second line of the statement. The first line is reserved for parameters that are passed to the utility.

■ If a table space is created explicitly, Analysis generates the command as shown in Figure 17 on page 273.

Figure 17: Example of -BMCC command

```
-BMCC 001700
COPY TABLESPACE BRDB101.N1
DEVT SYSDA COPYDDN (SYCL0001,SYCR0002)
```

■ If the table space is created implicitly, the character T precedes the command text. Execution looks up the table space name during processing and substitutes it for the table name before invoking the utility.

■ If DB2 created XML or LOB objects implicitly, the -BMCC command includes the following parameters:

—XMLTBO or LOBTBO, which identifies the owner of the table
—XMLTBN or LOBTBN, which identifies the name of the table
—XMLCOL or LOBCOL, which identifies the name of the XML or LOB column

The COPY command syntax also includes the name of the XML or LOB table space that supports the XML or LOB column in the base table.

For more information, see the BMC Next Generation Technology Copy for DB2 for z/OS Reference Manual.

-BMCD (BMC UNLOAD PLUS utility)

The -BMCD command invokes the UNLOAD PLUS utility.

The utility runs as a subtask. The -BMCD command can be processed in parallel. For more information about the worklist parallelism feature of the Database Administration and BMC Object Administration for DB2 solutions, see the ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2.

You can run the UNLOAD PLUS utility to perform the following functions:

■ Dynamically allocate unload data sets
■ Use fast unload
■ Unload from image copies
■ Use multitasking
- Unload converted data
- Migrate tables with ROWID columns
- Unload data in XML and LOB columns

For more information, see the *UNLOAD PLUS for DB2 Reference Manual*.

**Use of -BMCD to dynamically allocate unload data sets**

When you specify to use the UNLOAD PLUS and LOADPLUS utilities, the utilities can dynamically allocate the unload data sets.

When Analysis generates the -BMCD command, it specifies an OUTPUT descriptor for primary data sets. The DSNAME parameter in the descriptor specifies the fully qualified data set name for the object that is unloaded. Figure 18 on page 274 shows an example of a -BMCD command that uses dynamic allocation.

**Figure 18: -BMCD command—dynamic allocation of data sets**

```
-BMCD 000400
UNLOAD UNLOADDN (SYSREC01)
ACTIVE (YES,NO)
ON FAILURE ALL TERMINATE UTILITY
DELETEFILES YES
DISCARDS 1
NULLTYPE T1 NULLCHAR ?
SELECT *
FROM   ACMU02.T_U24PV
ORDER BY COLUMN_1
;
OUTPUT SYSREC01
UNIT SYSDA
DSNAME 'RDAJZB3.DUNLDA4.ACMU02.U24PV.P'
```

The data set name that is used for an object for both UNLOAD PLUS and LOADPLUS should be the same in the worklist. Do not modify the name of the data set in the DSNAME parameter for an object without modifying the name of the corresponding data set in the INDSN parameter for the object.

When you use the UNLOAD PLUS utility to dynamically allocate unload data sets, Analysis generates the following parameters for the -BMCD command:

```
ON FAILURE ALL TERMINATE UTILITY
DELETEFILES YES
```

If UNLOAD PLUS cannot unload the table, these parameters cause UNLOAD PLUS to delete the dynamically allocated unload (SYSREC) data set.
Use of -BMCD to perform a fast unload

When you specify to use the UNLOAD PLUS and LOADPLUS utilities, the utilities can quickly unload data from one table and load it into another table that has a similar structure.

The FORMAT BMCLOAD parameter specifies to move the data, as shown in Figure 19 on page 275.

**Figure 19: -BMCD command—FORMAT BMCLOAD**

```
-BMCD 000350
    UNLOAD UNLOADDN (SYSR1001)
    ACTIVE (YES,NO)
    DISCARDS 1
    NULLTYPE T1 NULLCHAR ?
    FORMAT BMCLOAD
    SELECT *
    FROM   ACMU02.T_U20NV
    ;
```

*Note*  
The fast unload feature requires the CCSID definition of the table space to match the default CCSID definition of the subsystem that is specified in the DSNHDECP load module. If the definitions do not match, the product does not use the feature to unload and load data.

Use of -BMCD to unload from image copies

When you specify to use the UNLOAD PLUS utility, the SHRLEVEL REFERENCE control card specifies to unload from image copies that are marked as SHRLEVEL REFERENCE. (If a SHRLEVEL CHANGE control card was included instead, it would specify to unload from image copies that were marked as SHRLEVEL REFERENCE or SHRLEVEL CHANGE.)

The NULLTYPE T1 NULLCHAR ? control card explicitly specifies the UNLOAD PLUS installation options. Figure 20 on page 275 shows an example of the -BMCD command with the SHRLEVEL REFERENCE control card and NULLTYPE T1 NULLCHAR ? control card for a single-table table space.

**Figure 20: -BMCD command—SHRLEVEL REFERENCE**

```
-BMCD 003200
    UNLOAD UNLOADDN (SYSR1001)
    ACTIVE (YES,NO)
    SHRLEVEL REFERENCE INFILE BMCD1001 FULL
    DISCARDS 1
    NULLTYPE T1 NULLCHAR ?
    SELECT *
    FROM TEST.EMPLS
    ORDER BY NAME
    ;
```
Use of -BMCD to perform multitasking

When you specify to use the UNLOAD PLUS and LOADPLUS utilities, the utilities can multitask the unloading and loading of data.

For multitasking to occur, a partitioned table space must be specified. When multitasking is used, the R1001 ddname prefix is used with the UNLOADDN control card, as shown in Figure 21 on page 276. The number in the prefix is incremented for each partitioned table space in the worklist. In addition, yyyy is appended to the ddname for each partition in the table space.

Figure 21: -BMCD command—partitioned table space

When you unload and load data in partitioned table spaces that contain over 256 partitions, the product requires multitasking. The product determines the number of dynamically allocated unload (SYSREC) data sets that can be used in a single -BMCD command by the value specified in the MAXSYSREC keyword in the ALUIN input stream. The default value of the MAXSYSREC keyword is 256. Once the product determines the number of data sets, it divides the partitions to be unloaded among several -BMCD commands. For example, if the value of MAXSYSREC is 256 and the number of partitions in the table space is 540, Analysis generates three -BMCD commands:

- The first command unloads or loads the first 256 partitions.
- The second command unloads or loads the next 256 partitions.
- The third command unloads or loads the last 28 partitions.

Figure 22 on page 276 shows an example of the -BMCD commands for a worklist. The IPPARTS parameter in the JCLP command specifies the number of partitions for each -BMCD command.

Figure 22: -BMCD command—multitasking more than 256 partitions
Use of -BMCD to unload converted data

Analysis invokes the UNLOAD PLUS utility twice if the LOADPLUS utility is specified and data conversions are occurring.

In the first invocation, the utility uses a SELECT INTO statement for the columns in which data is converted to ensure that only valid (supported) conversions occur on the load. The data is written to a DUMMY DD; this data is not saved. In the second invocation, the data is unloaded. Figure 23 on page 277 shows an example of the -BMCD commands that are used to unload a single-table table space for a full-recovery baseline.

Figure 23: -BMCD command—data conversion in a single-table table space
INTO TB3COL1 CHAR(9) TRUNCATE
FROM TEST.JFLVERTB3
-
BMCD 000450
 UNLOAD UNLOADDN (BLRP0001)
  ACTIVE (YES,NO)
  NULLTYPE T1 NULLCHAR ?
  SELECT *
 FROM TEST.JFLVERTB3
  ORDER BY TB3COL1
-

**Note**
If you are restarting a worklist that contains both -BMCD and -UNRC commands, ensure that both commands use identical data set names, because these names are being recorded in a full-recovery baseline. If you must restart -BMCD with a different data set name, modify -UNRC to record the same data set name.

Figure 24 on page 278 shows an example of the -BMCD commands that are used for a multi-table table space.

**Figure 24: -BMCD command—data conversion in a multi-table table space**

```
-BMCD 000650
 UNLOAD UNLOADDN (DUMMY)
  ACTIVE (YES,NO)
  NULLTYPE T1 NULLCHAR ?
  SELECT TB2COL1
 INTO TB2COL1 CHAR(9) TRUNCATE
 FROM TEST.JFLVERTB2
 ;
  SELECT TB1COL1
 INTO TB1COL1 CHAR(9) TRUNCATE
 FROM TEST.JFLVERTB1
 ;
-BMCD 000700
 UNLOAD UNLOADDN (SYSR1001)
  ACTIVE (YES,NO)
  NULLTYPE T1 NULLCHAR ?
  SELECT '0001', TB2COL1, TB2COL2
 INTO BMCCONS_RECORDID CHAR(4),
 TB2COL1 ,
 TB2COL2
 FROM TEST.JFLVERTB2
 ORDER BY TB2COL1
 ;
  SELECT '0002', TB1COL1, TB1COL2
 INTO BMCCONS_RECORDID CHAR(4),
 TB1COL1 ,
 TB1COL2
 FROM TEST.JFLVERTB1
 ORDER BY TB1COL1
 ;
```

**Use of -BMCD to migrate tables with ROWID columns**

You can modify or migrate tables that contain ROWID columns with data.

If the ROWID column is defined as GENERATED ALWAYS, ALTER and CHANGE MANAGER omit the ROWID column in the UNLOAD and LOAD statements. If the
ROWID column is defined as GENERATED BY DEFAULT, ALTER and CHANGE MANAGER include the ROWID column in the UNLOAD and LOAD statements. The column is unloaded last in the sequence of columns and is also loaded last, even though it might not be the last column in the table. When a table that contains a ROWID column is unloaded, Analysis lists all of the columns that are unloaded in the UNLOAD statement in the worklist.

Use of -BMCD to unload data in XML and LOB columns

When you use the UNLOAD PLUS utility to unload data contained in XML or LOB columns, the utility unloads the data to a file reference output data set; this data set must be dynamically allocated and on DASD.

In addition to the SYRC OUTPUT descriptor, the -BMCD command specifies an OUTPUT descriptor of SYxxnn for file reference data sets that contain LOB or XML data. The xx represents LB for LOB data or XC for XML data, and nn represents a sequential number for each LOB or XML column in the table. Each OUTPUT descriptor includes the directory blocks, and primary and secondary space. Analysis obtains these values from the JCL Generation product options file (POF).

Figure 25 on page 279 shows an example of a -BMCD command that unloads XML data.

Figure 25: -BMCD command—unloading XML data
-BMCK (BMC CHECK PLUS utility)

The -BMCK command invokes the CHECK DATA option of the CHECK PLUS utility, which checks referential constraints in table spaces.

The -BMCK command can be processed in parallel. For more information about the worklist parallelism feature of the Database Administration and BMC Object Administration for DB2 solutions, see the ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2.

If the table space is created explicitly, the command is executed as shown in Figure 26 on page 280.

**Figure 26: -BMCK command**

```
-BMCK 004200
  CHECK DATA TABLESPACE KIF8D22A.MAF8S22E
  ERRDDN SYSER001 WORKDDN SYSUT001
```

If the table space is created implicitly, the command text is preceded with the character T and the table space name is set to the table name. Execution looks up the table space name during processing and substitutes it for the table name before invoking the utility. This command might also be used due to referential integrity considerations.

For additional information, see the CHECK PLUS for DB2 Reference Manual.

-BMCL (BMC LOADPLUS utility)

The -BMCL command invokes the LOADPLUS utility.

This program runs as a subtask. The -BMCL command can be processed in parallel. For more information about the worklist parallelism feature of the Database...
Administration and BMC Object Administration for DB2 solutions, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

You can run the LOADPLUS utility to perform the following functions:

- Use fast load
- Use multiple work data sets
- Register image copies
- Create partition-level image copies
- Use multitasking
- Migrate tables with ROWID columns
- Load data in XML and LOB columns
- Delete dynamically allocated discard data sets
- Migrate only data

**Use of -BMCL to perform a fast load**

If you specify to use the UNLOAD PLUS and LOADPLUS utilities, the utilities can quickly unload data from one table and load it into another table that has a similar structure.

The FORMAT BMCUNLOAD parameter specifies to move the data, as shown in Figure 27 on page 281.

**Figure 27: -BMCL command—FORMAT BMCUNLOAD**

```plaintext
-BMCL 003950
  LOAD DATA REPLACE
  INDDN SYSR1001
  ERRDDN SYSR001
  DISCARDS 0
  DISCARDDDN SYSD1001
  DELETFILES NO
  IDENTITYOVERRIDE YES UPDATEMAXA YES
  FORMAT BMCUNLOAD
  WORKDDN SYST001
  LOADDDN SORT0001
  REDDEFINE NO
  UNIQUEINTO YES
  INTO TABLE ACMU102.T_U2ONV
  WHEN TABLE = 1
```

**Note**

The fast load feature requires the CCSID definition of the table space to match the default CCSID definition of the subsystem that is specified in the DSNHDECP load module. If the definitions do not match, the product does not use the feature to unload and load data.
Use of -BMCL with multiple work data sets

You can specify to use multiple SYSUT temporary work data sets to enable LOADPLUS to build nonclustering indexes for a table concurrently.

If the number of indexes in a table is equivalent to or less than the maximum number of SYSUT data sets, ALTER and CHANGE MANAGER create one SYSUT data set for each nonclustering index. When you specify to use multiple SYSUT temporary work data sets, SUT is used as the prefix for the WORKDDN parameter in the -BMCL command (see Figure 28 on page 282 for an example of a table that is stored in a single-table table space). In addition, the-JCLP command precedes the -BMCL command in the worklist and indicates the number of SYSUT data sets that are generated in the JCL.

Figure 28: -BMCL command—multiple SYSUT data sets

```
-JCLP 002250 BMCL NUMIX 2
-BMCL 002300
   LOAD DATA REPLACE
   INDDN SYSR1001
   ERRDDN SYSER001
   DISCARDS 0
   DISCARDDN SYSD1001
   DELETEFILES NO
   IDENTITYOVERRIDE YES UPDATEMAXA YES
   WORKDDN (SUT)
   LOADDN SORT0001
   REDEFINE NO
   UNIQUEINTO YES
   INTO TABLE C7UTC1C.TB13A
   (COLUMN_1          POSITION(*) INTEGER, 
    COLUMN_2          POSITION(*) CHAR(12),
    COLUMN_3          POSITION(*) SMALLINT,
    COLUMN_4          POSITION(*) CHAR(7),
    COLUMN_5          POSITION(*) DECIMAL(11,2),
    COLUMN_6          POSITION(*) DATE EXTERNAL)
```

One SYSUT data set is used when the table contains only one index or when the number of nonclustering indexes for a table exceeds the number of work data sets that you specified as the maximum. SYSUT001 is used for the WORKDDN parameter in the -BMCL command, as shown in Figure 29 on page 282. The table in this example is stored in a multi-table table space.

Figure 29: -BMCL command—single SYSUT data set

```
-BMCL 002850
   LOAD DATA RESUME YES
   INDDN SYSR1003
   ERRDDN SYSER001
   DISCARDS 0
   DISCARDDN SYSD1003
   DELETEFILES NO
   IDENTITYOVERRIDE YES UPDATEMAXA YES
   WORKDDN SYSUT001
   LOADDN SORT0001
   UNIQUEINTO YES
   INTO TABLE C7UTC1C.TB14A
   (COLUMN_1          POSITION(*) INTEGER, 
    COLUMN_2          POSITION(*) CHAR(12),
    COLUMN_3          POSITION(*) SMALLINT,
    COLUMN_4          POSITION(*) CHAR(7),
    COLUMN_5          POSITION(*) DECIMAL(11,2),
    COLUMN_6          POSITION(*) DATE EXTERNAL)
```
Use of -BMCL to register image copies

If any of the four image copies (primary and backup copies for the local and remote sites) are registered, Analysis inserts the REGISTER DDName parameter.

Figure 30 on page 283 shows an example of the -BMCL worklist command with the REGISTER parameter. If none of the image copies are registered, Analysis inserts the REGISTER NONE parameter.

Figure 30: -BMCL command—REGISTER parameter

```
-BMCL 001350
  LOAD DATA  REPLACE
  INDDN SYSR1001
  ERRDDN SYSER001
  DISCARDS 0
  DISCARDSDN SYSD1001
  DELETEFILES NO
  IDENTITYOVERRIDE YES UPDATEMAXA YES
  WORDDDN SYSUTO01
  LOADDDN SORT0001
  REDEFINE NO
  UNIQUEINTO YES
  COPY YES
    REGISTER SYCL0001,SYCR0001
    COPYDDN (SYCL0001,SYCR0001)
  INTO TABLE ACMX01E.T_X01PS
(COLUMN_NEW           POSITION(*) INTEGER,
 COLUMN_2             POSITION(*) SMALLINT,
 COLUMN_3             POSITION(*) CHAR(12),
 COLUMN_4             POSITION(*) CHAR(7),
 COLUMN_5             POSITION(*) CHAR(4))
```

Use of -BMCL to create partition-level image copies

Analysis can generate parameters for a partition-level image copy of a partitioned table space or index in a database when you select to use LOADPLUS as the load utility.

Figure 31 on page 283 shows an example of the -BMCL command that is used when partition-level image copies are requested for a table space that has five partitions.

Figure 31: -BMCL command—partition-level image copies

```
-SPAC 003150 DDNAME COO1 BMCL TS C7PCA1.TS35P5
-JCLP 003200 BMCL IPPARTS 5
-JCLP 003250 BMCL NUMPARTS 5
-BMCL 003300
  LOAD DATA  REPLACE
  INDDN R1003
  ERRDDN SYSER001
  DISCARDS 0
(COLUMN_3             POSITION(*) SMALLINT,
 COLUMN_4             POSITION(*) CHAR(7),
 COLUMN_5             POSITION(*) DATE EXTERNAL)
```
Use of -BMCL to perform multitasking

The LOADPLUS utility can be used with the UNLOAD PLUS utility to multitask the unloading and loading of data.

When you unload and load data in partitioned table spaces that contain more partitions than the value of the MAXSYSREC keyword, the product requires multitasking. The product uses the value of the MAXSYSREC keyword in the ALUIN input stream to determine the number of dynamically allocated data sets to use in a single -BMCL command. The default value of the MAXSYSREC keyword is 256. The first -BMCL command contains the LOAD REPLACE parameter; the remaining -BMCL commands contain the LOAD RESUME YES parameter and the REPLACE parameter for each partition.

The LOAD statement in the -BMCL command will contain the ORDER YES parameter if the following conditions exist:

- The utility is multitasking the loading of data.
- The table space contains compressed partitions.
- The value of the MAXSYSREC keyword is less than the number of partitions in the table space or a value is not specified for MAXSYSREC.

Figure 32 on page 284 shows an example of the -BMCL commands for loading a table space that contains 540 partitions. The IPPARTS parameter in the -JCLP command specifies the number of partitions for each -BMCL command.

Figure 32: -BMCL command—multitasking more than 256 partitions

```
-JCLP 001600 BMCL IPPARTS 256
-BMCL 001700
  LOAD DATA REPLACE
  INDSN('RDABKH1.DEBF.BH0514A.SR000001.P001'
  , 'RDABKH1.DEBF.BH0514A.SR000001.P002'
  , 'RDABKH1.DEBF.BH0514A.SR000001.P003'
  , 'RDABK1.DEBF.BH0514A.SR000001.P004'
  , 'RDABKH1.DEBF.BH0514A.SR000001.P005')
```
ALTER and CHANGE MANAGER worklist commands

,...

`'RDABKH1.DEBF.BH0514A.SR000001.P256`

)`

ERRDDN SYSER001
DISCARDS 0
ENUMROWS AUTO
DDTYPE DISCARD
ACTIVE YES
IFALLOC FREE
DELETEFILES NO SYSDISC YES
IDENTITYOVERRIDE YES UPDATEMAXA YES
UNIT SYSDA
DSNPAT 'RDABKH1.BH0514A.B91S128A.TS11A.SD000001'
ORDER YES
EBCDIC CCSID(37,0,0)
WORKDDN (SUT)
LOADDN SORTP
REDEFINE NO
UNIQUEINTO YES
INTO TABLE B91S128A.TB11A
 PART 1:256

{ COLIN_1
  POSITION(*) INTEGER.

COLSI_2
  POSITION(*) SMALLINT.

COLCH_3
  POSITION(*) CHAR(12).

COLCH_4
  POSITION(*) CHAR(7).

COLCH_5
  POSITION(*) CHAR(4).

COLVC_6
  POSITION(*) VARCHAR.

COLTS_7
  POSITION(*) TIMESTAMP EXTERNAL.

COLDT_8
  POSITION(*) DATE EXTERNAL.

COLIN_9
  POSITION(*) INTEGER
      NULLIF NULL001='?',
      POSITION(*) CHAR(1),
      NULL001
      POSITION(*) CHAR(1).

COLDC_10
  POSITION(*) DECIMAL(9,2)
      NULLIF NULL002='?',
      POSITION(*) CHAR(1),
      NULL002
      POSITION(*) CHAR(1).

COLDC_11
  POSITION(*) DECIMAL(6,3)
      NULLIF NULL003='?',
      POSITION(*) CHAR(1).

-JCLP 001800 BMCL IPPARTS 256
-BMCL 001900

LOAD DATA RESUME YES
INDSN('RDABKH1.DEBF.BH0514A.SR000001.P257'
   , 'RDABKH1.DEBF.BH0514A.SR000001.P258'
   , 'RDABKH1.DEBF.BH0514A.SR000001.P259'
   , 'RDABKH1.DEBF.BH0514A.SR000001.P260'
   , 'RDABKH1.DEBF.BH0514A.SR000001.P261'
   ...
   , 'RDABKH1.DEBF.BH0514A.SR000001.P512'
)
)
ERRDDN SYSER001
DISCARDS 0
ENUMROWS AUTO
DDTYPE DISCARD
ACTIVE YES

Chapter 5  Worklist commands  285
IFALLOC FREE
DELETEFILES NO SYSDISC YES
IDENTITYOVERRIDE YES UPDATEMAXA YES
UNIT SYSDA
DSNPAT 'RDABKH1.BH0514A.B91S128A.TS11A.SD000002'
ORDER YES
EBCDIC CCSID(37,0,0)
WORKDDN (SUT)
LOADDDN SORTP
UNIQUEINTO YES
INTO TABLE B91S128A.TB11A
PART 257:512
REPLACE
(
   COLIN_1
   POSITION(*) INTEGER,
   COLSI_2
   POSITION(*) SMALLINT,
   COLCH_3
   POSITION(*) CHAR(12),
   COLCH_4
   POSITION(*) CHAR(7),
   COLCH_5
   POSITION(*) CHAR(4),
   COLVC_6
   POSITION(*) VARCHAR,
   COLTS_7
   POSITION(*) TIMESTAMP EXTERNAL,
   COLDT_8
   POSITION(*) DATE EXTERNAL,
   COLIN_9
   POSITION(*) INTEGER
   )
   NULLF NULL001='?',
   NULL001
   POSITION(*) CHAR(1),
   NULL002
   POSITION(*) DECIMAL(9,2)
   NULLF NULL002='?',
   NULL002
   POSITION(*) CHAR(1),
   NULL003
   POSITION(*) DECIMAL(6,3)
   NULLF NULL003='?',
   NULL003
   POSITION(*) CHAR(1))
-JCLP 002000 BMCL IPPARTS 28
-BMCL 002100
LOAD DATA RESUME YES
INDSN('RDABKH1.DEBF.BH0514A.SR000001.P513'
   'RDABKH1.DEBF.BH0514A.SR000001.P514'
   'RDABKH1.DEBF.BH0514A.SR000001.P515'
   'RDABKH1.DEBF.BH0514A.SR000001.P516'
   'RDABKH1.DEBF.BH0514A.SR000001.P517'
   'RDABKH1.DEBF.BH0514A.SR000001.P540'
   )
ERRDDN SYER001
DISCARDS 0
ENUMROWS AUTO
DDTYPE DISCARD
ACTIVE YES
IFALLOC FREE
DELETEFILES NO SYSDISC YES
IDENTITYOVERRIDE YES UPDATEMAXA YES
UNIT SYSDA
DSNPAT 'RDABKH1.BH0514A.B91S128A.TS11A.SD000003'
ORDER YES
EBCDIC CCSID(37,0,0)
WORKDDN (SUT)
Use of -BMCL to migrate tables with ROWID columns

You can modify or migrate tables that contain ROWID columns with data.

If the ROWID column is defined as GENERATED ALWAYS, ALTER and CHANGE MANAGER omit the ROWID column in the UNLOAD and LOAD statements. If the ROWID column is defined as GENERATED BY DEFAULT, ALTER and CHANGE MANAGER include the ROWID column in the UNLOAD and LOAD statements. The column is unloaded last in the sequence of columns and is also loaded last, even though it might not be the last column in the table. When a table that contains a ROWID column is unloaded, Analysis lists all of the columns that are unloaded in the UNLOAD statement in the worklist.

Use of -BMCL to load data in XML and LOB columns

When you use the LOADPLUS utility to unload data contained in XML or LOB columns, the utility loads the data from a file reference input data set that contains the data.

When the -BMCL command contains the LOAD DATA REPLACE parameter, Analysis adds the PRELOAD LOAD parameter. If the -BMCL command contains the LOAD DATA RESUME YES parameter and the number of discards is greater than...
zero, Analysis adds the PRELOAD CONTINUE parameter. Figure 33 on page 288 shows an example of a -BMCL command that loads XML data.

**Figure 33: -BMCL command—loading XML data**

```plaintext
-BMCL 001300
LOAD DATA REPLACE
INDSN('RDACRJ.XML_SYSREC.SR000001')
ERRDDN SYSER001
PRELOAD LOAD
DISCARDS 1
DISCARDDN SYSDS001
DELETEFILES NO
EBCDIC CCSID(37,0,0)
WORKDDN (SUT)
LOADDN SORT0001
ENFORCE NO
UNIQUEINTO YES
INTO TABLE C9AXM1.TBMMS03A
(
  TABKEY
  POSITION(*) INTEGER,
  CHAR5
  POSITION(*) CHAR(5)
  NULL001
  POSITION(*) CHAR(1),
  CHAR10
  POSITION(*) CHAR(10)
  NULL002
  POSITION(*) CHAR(1),
  DEC15
  POSITION(*) DECIMAL(15,0)
  NULL003
  POSITION(*) CHAR(1),
  DEC15_3
  POSITION(*) DECIMAL(15,3)
  NULL004
  POSITION(*) CHAR(1),
  VAR_CHAR15
  POSITION(*) VARCHAR
  NULL005
  POSITION(*) CHAR(1),
  XMLCOL1
  POSITION(*) VARCHAR CLOBF
  NULL006
  POSITION(*) CHAR(1),
  XMLCOL2
  POSITION(*) VARCHAR CLOBF,
  TIMESTAMP
  POSITION(*) TIMESTAMP EXTERNAL)
```

For an example of a worklist in which LOB data is unloaded and loaded, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

**Use of -BMCL to delete dynamically allocated discard data sets**

When you specify to use the UNLOAD PLUS and LOADPLUS utilities and to dynamically allocate unload data sets, the LOADPLUS utility also dynamically allocates a discard data set for each LOAD statement, if needed.
The LOADPLUS utility does not delete the discard data set automatically, even if the data set is empty. The Analysis component adds the SYSDISC YES option to the DELETEFILES NO parameter in the -BMCL worklist command, as shown in Figure 34 on page 289. The SYSDISC YES option instructs LOADPLUS to delete the discard data set if the data set is empty.

**Figure 34: -BMCL command—deleting dynamically allocated discard data sets**

```
-BMCL 004050
LOAD DATA REPLACE
INDSN('RDACRJ.LOBDATA.SR000001')
ERRDDN SYSER001
DISCARDS 0
ENUMROWS (0,0,0)
DDTYPE DISCARD
ACTIVE YES
IFALLOC FREE
DELETEFILES NO SYSDISC YES
IDENTIVITYOVER ride YES UPDATEMAXA YES
UNIT SYSDA
DSNPAT 'RDACRJ.LOBDATA.SD000001'
WORKDDN SYSUT001
LOADDN SORT0001
REDEFINE NO
UNIQUEINTO YES
INTO TABLE RDACRJ.TEMO
```

ALTER and CHANGE MANAGER perform the actions cited below if you do one of the following:

- You do not specify to dynamically allocate the unload (SYSREC) data set in the Analysis override options
- You specify to dynamically allocate the unload (SYSREC) data set and you manually add the DISCARDS 1 keyword to the ALUIN input stream (or you specify DISCARDS=(1,R), in the installation options)

The resulting actions are as follows:

- When generating the worklist, the Analysis component omits the SYSDISC YES option in the DELETEFILES NO parameter in the -BMCL worklist command.
- If DISCARDS is set to 1, the JCL Generation component allocates the discard data set in the JCL that it generates for the worklist.
- When the Execution component runs the worklist, the LOADPLUS utility does not delete the discard data set.

**Use of -BMCL to migrate only data with FORCELOADREPLACE**

When you specify the FORCELOADREPLACE keyword in the ALUIN input stream, Analysis generates one of a few forms of the LOAD DATA syntax.
Analysis generates the following forms of the syntax:

- **LOAD DATA RESUME YES INTO TABLE**
  Analysis uses this syntax when you are migrating only data from a single table at the table level of a segmented table space (Figure 35 on page 290). Analysis does not assume that the structures on the sending and receiving subsystems are identical. The load utility deletes the existing data in the table before loading.

  **Figure 35: BMCL command—LOAD DATA RESUME YES INTO TABLE**

  ```
  SQL 000850 DELETE FROM J1ALLN1.T_T02SSEG
  SYNC 000900 END OF SQL DELETE FROM TABLES SECTION
  JCLP 000950 BMCL NUMIX 2
  BMCL 001000
    LOAD DATA RESUME YES
    INDSN('RDAJ2ZB4.DEGA.J120824G.SR000001')
    ERRDDN SYSER001
    PRELOAD CONTINUE
    DISCARDS 1
    DISCARDDN SYSDS001
    DELETEFILES NO
    ORDER YES
    EBCDIC CCSID(37,0,0)
    WORKDDN (SUT)
    LOADDN SORT0001
    ENFORCE NO
    UNIQUE INTO YES
    INTO TABLE J1ALLN1.T_T02SSEG
    (COLIN_1 POSITION(*) INTEGER,
     COLSI_2 POSITION(*) SMALLINT,
     COLCH_3 POSITION(*) CHAR(20),
     COLBI_4 POSITION(*) BINARY(200),
     COLDC_5 POSITION(*) DECIMAL(31,20),
     COLDC_6 POSITION(*) DECIMAL(11,2),
     COLFL_7 POSITION(*) FLOAT(21),
     COLFL_8 POSITION(*) FLOAT(53),
     COLDT_9 POSITION(*) DATE EXTERNAL,
     COLTM_10 POSITION(*) TIME EXTERNAL,
     BMCSKIPTMSTAMP001 POSITION(*) CHAR(26))
  SYNC 001050 LOAD TB J1ALLN1.T_T02SSEG COMPLETE
  ```

- **LOAD DATA RESUME YES INTO TABLE REPLACE**
  Analysis uses this syntax when you are migrating only data from one or more tables at the table space level of a segmented table space (Figure 36 on page 291). Analysis assumes that the structures on the sending and receiving subsystems are identical, and generates the LOAD statements based on the structure of the
receiving subsystem. The load utility replaces the data in the table; an SQL DELETE statement is not needed.

Figure 36: -BMCL command—LOAD DATA RESUME YES INTO TABLE REPLACE

```
* -SQL DELETE NOT NEEDED - J1DYNR4.T_T02SSEG
* -SYNC 001850 END OF SQL DELETE FROM TABLES SECTION
* -SYNC 001900 LOAD TB J1DYNR4.T_T01ICP COMPLETE
* -JCLP 001950 BMCL NUMIX 2
- BMCL 002000
  LOAD DATA  RESUME YES
  INDSN('RDAJZB4.DEGA.J120727D.SR000002')
  ERRDDN SYSER001
  PRELOAD CONTINUE
  DISCARDS 1
  DISCARDON SYSDS001
  DELETEFILES NO
  ORDER YES
  FORMAT BMCUNLOAD
  WORKDDN (SUT)
  LOADDN SORT0001
  ENFORCE NO
  UNIQUEINTO YES
  INTO TABLE J1DYNR4.T_T02SSEG
  REPLACE
  WHEN TABLE = 1
* -SYNC 002050 LOAD TB J1DYNR4.T_T02SSEG COMPLETE
```

**LOAD DATA REPLACE INTO TABLE**

Analysis uses this syntax when you are migrating only data from an entire table space at the table space level of a partitioned table space (Figure 37 on page 291).

Analysis assumes that the structures on the sending and receiving subsystems are identical, and generates the LOAD statements based on the structure of the receiving subsystem. The load utility replaces the data in the table space.

Figure 37: -BMCL command—LOAD DATA REPLACE INTO TABLE

```
-JCLP 001900 BMCL NUMIX 2
-JCLP 001950 BMCL IPPARTS 3
- BMCL 002000
  LOAD DATA  REPLACE
  INDSN('RDAJZB4.DEGA.J120727D.SR000001.P001',
       'RDAJZB4.DEGA.J120727D.SR000001.P002',
       'RDAJZB4.DEGA.J120727D.SR000001.P003')
  ERRDDN SYSER001
  DISCARDS 1
  DISCARDON SYSDS001
  DELETEFILES NO
  ORDER YES
  FORMAT BMCUNLOAD
  WORKDDN (SUT)
  LOADDN SORTP
  ENFORCE NO
  UNDEFINE NO
  UNIQUEINTO YES
  INTO TABLE J1DYNR4.T_T01ICP
  WHEN TABLE = 1
  -SYNC 002050 LOAD TB J1DYNR4.T_T01ICP COMPLETE
```
Notes for -BMCL

For some objects, Analysis generates the -BMCL command differently.

Note the following items when the -BMCL command is used in a worklist:

- The utility command starts on the second line of the statement. The first line is reserved for parameters that are passed to the utility.

- If a table space is being created in the worklist, Analysis includes the REDEFINE NO parameter in the -BMCL command.

- When using LOADPLUS in a worklist environment, your work files might not be dynamically allocated automatically, even if you have specified ACTIVE=YES in your utility installation options module.

  If you are operating in a worklist environment, LOADPLUS ignores the value that you specified in the DELFILES utility installation option and always uses DELFILES=NO. To delete your work files, you must edit your worklist and change DELFILES NO to DELFILES YES in the LOAD command. However, BMC recommends that you do not delete your work files because subsequent steps in the worklist might require the use of the files. Instead, use the JCL Generation Static Data Set Options panel to specify to use a data set cleanup step. (For more information, see the ALTER and CHANGE MANAGER for DB2 User Guide, Volume 1.)

- If you specify to collect statistics with other utilities (such as the LOADPLUS utility), and you specify to use the BMCSTATS utility to update statistics, Analysis includes the BMCSTATS YES parameter in the -BMCL command. When the command includes the BMCSTATS YES parameter, generated messages are output to SYSPRINT. The REPORT NO parameter, which suppresses the creation of a report of updated statistics to output to SYSPRINT, is available only when the BMCSTATS utility is run as a stand-alone utility (with the -BMCS command). BMCSTATS does not generate a statistics report when other utilities collect statistics.

- If you are converting from a DECIMAL column with a scale of 0 to a CHAR or VARCHAR column, Analysis includes the REMOVEDECIMAL parameter on the column in the -BMCL command. The REMOVEDECIMAL parameter instructs LOADPLUS to remove the fractional portion, including the decimal point, of the value that it loads.

- Unique values that are assigned to rows in an identity column might change when the following conditions exist:

  - You have defined an identity column in your table with GENERATED ALWAYS.
The table is dropped and re-created, and the values are reloaded. To have LOADPLUS preserve the existing values in the identity column, simply run Analysis and Execution.

By default, the ALUIN input stream includes the NOREGENIDENTITY keyword. NOREGENIDENTITY instructs Analysis not to allow DB2 to generate new values for an identity column that is defined as GENERATED ALWAYS if the table is dropped and re-created. In addition, Analysis adds the IDENTITYOVERRIDE YES and UPDATEMAXA YES parameters to the -BMCL (LOADPLUS) worklist command. IDENTITYOVERRIDE YES tells LOADPLUS to load identity column values from the input file. UPDATEMAXA YES tells LOADPLUS to update the MAXASSIGNEDVAL column of SYSIBM.SYSSEQUENCES when loading identity column values from an input file.

For more information about LOADPLUS, see the LOADPLUS for DB2 Reference Manual.

-BMCR (BMC REORG PLUS utility)

The -BMCR command invokes the REORG PLUS utility.

This program runs as a subtask. You can run the REORG PLUS utility to perform the following functions:

- Use multiple work data sets
- Register image copies
- Create partition-level image copies
- Perform an online reorg (SHRLEVEL CHANGE)

In addition, REORG PLUS can use dynamically allocated data sets.

Use of -BMCR with multiple work data sets

You can specify to use multiple SYSUT temporary work data sets to improve I/O performance when more than one nonclustering index exists in a table.

If the number of nonclustering indexes in a table is equivalent to or less than the maximum number of SYSUT data sets, ALTER and CHANGE MANAGER create one SYSUT data set for each nonclustering index. When you specify to use multiple SYSUT temporary work data sets, SUT is used as the prefix for the WORKDDN parameter in the -BMCR command (see Figure 38 on page 294). In addition, the
-JCLP command precedes the -BMCR command and indicates the number of SYSUT data sets that are generated in the JCL.

**Figure 38: -BMCR command—multiple SYSUT data sets**

```
-JCLP 000600 BMCR NUMIX   3
-BMCR 000650
  REORG TABLESPACE C7UTE1.TS21M
  DELETEFILES NO
  UNLDDN SYR1001
  WORKDDN (SUT)
  SORTDATA LOG NO
```

For table spaces that are VCAT defined and thus are reorganized in two phases, you cannot use multiple SYSUT data sets. WRK1001 n is used as the ddname for the WORKDDN parameter in the -BMCR command, as shown in **Figure 39 on page 294**.

**Figure 39: -BMCR command—two-phase reorg**

```
-BMCR 000550
  REORG TABLESPACE C7UTE1.TS32V2
  UNLOAD PAUSE
  DELETEFILES NO
  UNLDDN SYR1001
  WORKDDN (WRK1001)
  SORTDATA LOG NO
```

**Use of -BMCR to register image copies**

If any of the four image copies (primary and backup copies for the local and remote sites) are registered, Analysis inserts the REGISTER ddname parameter.

**Figure 40 on page 294** shows an example of the -BMCR worklist command with the REGISTER parameter. If none of the image copies are registered, Analysis inserts the REGISTER NONE parameter.

**Figure 40: -BMCR command—REGISTER parameter**

```
-BMCR 000350
  REORG TABLESPACE ACMX02.X23PV
  UNLOAD PAUSE
  DELETEFILES NO
  UNLDDN SYR1001
  WORKDDN (WRK1001)
  COPY YES
    REGISTER E001
    COPYDDN (C001)
    RECOVERYDDN (E001)
  SORTDATA LOG NO
```
Use of -BMCR to create partition-level image copies

Analysis can generate parameters for a partition-level image copy of a partitioned table space or index in a database when you select to use REORG PLUS as the reorg utility.

Figure 41 on page 295 shows an example of the REORG PLUS utility being used to reorganize a subset of partitioned table spaces in a database and to copy only those partitions that are being reorganized.

Figure 41: -BMCR command—image copies of some partitions

-JCLP D00650 BMCR COPYPARTS
  -BMCR D00700
  REORG TABLESPACE C7PCB1.TS35P5
  DELTEFILES NO
  UNLDDN SYRI001
  WORKDDN (WRK1001)
  PART 3
  COPY YES
  REGISTER C001
  COPYDDN (C001)
  SORTDATA LOG NO

Figure 42 on page 295 shows an example of the REORG PLUS utility being used to create partition-level image copies of all of the partitions of the table space.

Figure 42: -BMCR command—image copies of all partitions

-JCLP D00600 BMCR NUMPARTS 5
  -BMCR D00650
  REORG TABLESPACE C7PCB1.TS35P5
  DELTEFILES NO
  UNLDDN SYRI001
  WORKDDN (WRK1001)
  COPY YES
  REGISTER C001
  COPYDDN (C001)
  SORTDATA LOG NO

Use of -BMCR to perform an online reorg

In some cases, the only changes that you make in a work ID can be accomplished by using ALTER statements and online reorganizations (for example, online schema changes).

In those cases, you can use REORG PLUS to perform an online reorg (SHRLEVEL CHANGE) of your table spaces. Figure 43 on page 295 shows an example of the parameters that the REORG PLUS utility uses. The utility creates an inline image copy when it reloads the table space.

Figure 43: -BMCR command—online reorg

-BMCR D000500
  REORG TABLESPACE R1ALL.T05PBR
Note

When you perform an online REORG using REORG PLUS, Analysis creates an inline image copy by partition and the following message is displayed during the execution:

```
BMC396307I PARTCOPY TURNED ON TO SUPPORT BMCREORG AND NODYNREORG
```

Use of -BMCR to dynamically allocate data sets

ALTER and CHANGE MANAGER support the dynamic allocation of data sets for REORG PLUS.

When the Analysis component generates the worklist, the component includes the names of the data sets. You can enable or disable dynamic allocation by using the DYNREORG installation option, the DYNREORG or NODYNREORG ALUIN keywords, or an option on the Analysis Utility Dataset Options panel. Figure 44 on page 296 shows an example of the syntax that REORG PLUS uses for the data sets.

Figure 44: -BMCR command—dynamically allocated data sets

```
-BMCR 000550
REORG TABLESPACE CJBASIC.T02ASSEG
UNLOAD CONTINUE
DELETEFILES NO

DDTYPE UNLOAD     ACTIVE YES
ALLOC ANY
IFALLOC USE
UNIT  SYSDA
DSNPAT 'RDACRJ.DEGA.RB10206D.SR000001.P&PART'
UNLDDN SYRC

DDTYPE WORK     ACTIVE YES
ALLOC ANY
IFALLOC USE
UNIT  SYSDA
DSNPAT 'RDACRJ.RB10206D.STEP1.SU000001.&DDNAME'
WORKDDN DUT

DDTYPE LOCPFCPY     ACTIVE YES
ALLOC ANY
IFALLOC USE
UNIT  SYSDA
DSNPAT 'RDACRJ.RB10206D.T02ASSEG.P000.LP0001'
COPY YES
INLINE YES
COPYDDN (DCPYLOCP)
SORTDATA LOG NO
```
Notes for -BMCR

For some objects, Analysis generates the -BMCR command differently.

Note the following items when the -BMCR command is used in a worklist:

- The utility command starts on the second line of the statement. The first line is reserved for parameters that are passed to the utility.

- When using REORG PLUS in a worklist environment, your work files might not be dynamically allocated automatically, even if you have specified ACTIVE=YES in your utility installation options module.

  If you are operating in a worklist environment, REORG PLUS ignores the value that you specified in the DELFILES utility installation option and always uses DELFILES=NO. To delete your work files, you must edit your worklist and change DELETEFILES NO to DELETEFILES YES in the REORG command.

  However, BMC recommends that you do not delete your work files because subsequent steps in the worklist might require the use of the files. Instead, use the JCL Generation Static Data Set Options panel to specify to use a data set cleanup step. (For more information, see the ALTER and CHANGE MANAGER for DB2 User Guide, Volume 1.)

- If you specify to collect statistics with other utilities (such as the REORG PLUS utility), and you specify to use the BMCSTATS utility to update statistics, Analysis includes the BMCSTATS YES parameter in the -BMCR command. When the command includes the BMCSTATS YES parameter, generated messages are output to SYSPRINT. The REPORT NO parameter, which suppresses the creation of a report of updated statistics to output to SYSPRINT, is available only when the BMCSTATS utility is run as a stand-alone utility (with the -BMCR command). BMCSTATS does not generate a statistics report when other utilities collect statistics.

  For more information about REORG PLUS, see the REORG PLUS for DB2 Reference Manual.

-BMCS (BMC BMCSTATS utility)

The -BMCS command invokes the DASD MANAGER PLUS BMCSTATS utility.

The -BMCS command can be processed in parallel. For more information about the worklist parallelism feature of the Database Administration or BMC Object
Administration for DB2 solutions, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

Note the following items when the -BMCS command is used in a worklist:

- The utility command starts on the second line of the statement. The first line is reserved for parameters that are passed to the utility, such as NEW, TERM, MAINT, NEW/RESET, and TERM/RESET.

- If a table space has been rebuilt or has tables that have been rebuilt or added, the Analysis component creates a -BMCS command for the table space. If the table space is created explicitly, the command is executed as shown in Figure 45 on page 298.

  ![Figure 45: -BMCS command](image)

- If the table space is created implicitly, the command text is preceded with the character T and the table space name is set to the table name. Execution looks up the table space name during processing and substitutes it for the table name before invoking the utility. This command might also be used due to referential integrity considerations.

- If an index is created or rebuilt for a table space that had no table changes, Analysis creates a -BMCS command for the index.

- By default, the messages that are generated to report the collected statistics are not output to SYSPRINT.

- The BMCSTATS utility requires several data sets. For a list of the data sets, see the *DASD MANAGER PLUS for DB2 Reference Manual*.

**-BMCV (NGT Recover utility)**

The -BMCV command invokes the NGT Recover utility.

This command and its parameters are passed to NGT Recover without any processing. You can run the NGT Recover utility to perform the following functions:

- Rebuild indexes
- Load from image copies
- Dynamically allocate copy data sets
- Register image copies
- Create partition-level image copies
- Migrate image copy data

**Use of -BMCV to rebuild indexes**

The -BMCV command can invoke NGT Recover to rebuild an index.

The command executes as shown in Figure 46 on page 299.

**Figure 46: -BMCV command—rebuild index**

```
-BMCV 000350
   REBUILD INDEX (RDAJZB.IXNS1CNO)
   NOWORKDDN
   REDEFINE NO
```

*Note*

If the PIECESIZE attribute is altered and an index is rebuilt, the REDEFINE NO parameter is not used in the -BMCV command.

If a cloned table exists for a base table, and an index is being created for the base table, new indexes on tables with clones are marked DEFER YES in the create statement, and the CLONE keyword is added to the -BMCV REBUILD INDEX statement, as shown in Figure 47 on page 299.

**Figure 47: -BMCV command—rebuild index clone**

```
-BMCV 000450
   REBUILD INDEX (RJCLONE3.I_T02PBG)
   CLONE
   NOWORKDDN
   REDEFINE NO
```

**Use of -BMCV to load from image copies**

The -BMCV command can invoke NGT Recover to migrate data by using an image copy.

The command executes as shown in Figure 48 on page 299.

**Figure 48: -BMCV command—NGT Recover copy**

```
-BMCV 000760
   RECOVER TABLESPACE JFLDB01.JFLPT02A
   DSNUM 2
   INCOPY FULL
   DSNAME JFL.JFLPT02.PT2
   TOCOPY LASTCOPY
   RECOVER TABLESPACE JFLDB01.JFLPT02A
   DSNUM 1
   OBIDXLAT RESET DBID (276,)
```
Use of -BMCV to dynamically allocate copy data sets

The NGT Recover utility can dynamically allocate the data sets that are used to create valid image copies.

JCL Generation uses the symbolic variable that is specified in the JCL Generation Data Set Options for Copies panels to create an OUTPUT descriptor for Analysis. When generating the -BMCV command, Analysis specifies an OUTPUT descriptor for local and recovery primary and backup copies. The DSNAME parameter in the descriptor specifies the data set name for the object that is copied. Figure 49 on page 300 shows an example of a -BMCV command that uses dynamic allocation.

Figure 49: -BMCV command—dynamic allocation of data sets

```jcl
-BMCV 000850
  OUTPUT DCPYLOCP
    UNIT CART
    DSNAME RDAMCG.&UTIL.A.&TS
    STACK YES
    VOLCNT 99
  OUTPUT DCPYLOCB
    UNIT CART
    DSNAME RDAMCG.&UTIL.B.&TS
    STACK YES
    VOLCNT 99
  OUTPUT DCPYREMP
    UNIT CART
    DSNAME RDAMCG.&UTIL.C.&TS
    STACK YES
    VOLCNT 99
  RECOVER TABLESPACE ACMX01.X05SSJ
    OBIDXLAT RESET DBID (3209,)
    PSID (34,)
    OBID (35,)
    INCOPY FULL
    DSNNAME RDAMCG.DEAH.ACMX01.X05SS.D0418A
    OUTCOPY YES
    REGISTER DCPYLOCP,DCPYLOCB,DCPYREMP
    OUTCOPYDDN (DCPYLOCP,DCPYLOCB)
    RECOVERYDDN (DCPYREMP)
    TOCOPY LASTCOPY
    REBUILD INDEX(ALL) TABLESPACE ACMX01.X05SSJ
    NOWORKDDN
    DBID 3209,NEWDB=ACMX01
    PSID 34,NEWTS=X05SSJ
    OBID 35,NEWTB=ACMX01.T_X05SSJ
```

If the table spaces or indexes are partitioned, the P&PART symbolic variable is used in the DSNAME parameter to ensure that the data set name for each object is unique for partition-level image copies.
If you select to use NGT Recover to dynamically allocate copy data sets and to build image copies, the data sets cannot be stacked.

**Use of -BMCV to register image copies**

If *any* of the four image copies (primary and backup copies for the local and remote sites) are registered, Analysis inserts the REGISTER *DDName* parameter.

“Use of -BMCV to dynamically allocate copy data sets” on page 300 shows an example of the -BMCV worklist command with the REGISTER parameter. If *none* of the image copies are registered, Analysis inserts the REGISTER NONE parameter.

**Use of -BMCV to create partition-level image copies**

Analysis can generate parameters for a partition-level image copy of a partitioned table space or index in a database when you select to use NGT Recover.

Figure 50 on page 301 shows an example of the -BMCV command that is used when partition-level image copies are requested for a database that contains a table space that has four partitions. The OPTION parameter indicates that the utility is copying the table space by partition.

**Figure 50: -BMCV command—partition-level image copies**

```
-Spac 002250  DDName C001  BMCV  TS  C7PCA1.TS36P4
-JCLP 002300  BMCV  NUMPARTS 4
-BMCV 002350
   Option OUTCOPY BYPART
   RECOVER TABLESPACE C7PCC1.TS36P4
   DSNUM 4
   INCOPY FULL
   DSNAMe RDACEL.CPIDEBA.C7PCA1.TS34P4.P04.D0613B
   OUTCOPY YES
        REGISTER C001
        OUTCOPYDDN (C001)
        TOCOPY LASTCOPY
   RECOVER TABLESPACE C7PCC1.TS36P4
   DSNUM 3
   INCOPY FULL
   DSNAMe RDACEL.CPIDEBA.C7PCA1.TS33P3.P03.D0613B
   OUTCOPY YES
        REGISTER C001
        OUTCOPYDDN (C001)
        TOCOPY LASTCOPY
   RECOVER TABLESPACE C7PCC1.TS36P4
   DSNUM 2
   INCOPY FULL
   DSNAMe RDACEL.CPIDEBA.C7PCA1.TS33P3.P02.D0613B
   OUTCOPY YES
        REGISTER C001
        OUTCOPYDDN (C001)
        TOCOPY LASTCOPY
   RECOVER TABLESPACE C7PCC1.TS36P4
   DSNUM 1
   OBI DXLAT RESET DBID (1612.)
```
For more information, see the *BMC Next Generation Technology Recover for DB2 for z/OS Reference Manual*.

### Use of -BMCV to migrate image copy data

The NGT Copy EXPORT command and NGT Recover IMPORT command enable you to migrate an image copy or set of image copies within a DB2 subsystem or to another DB2 subsystem.

The EXPORT and IMPORT commands comprise the Copy Migration feature of the following solutions:

- Recovery Management
- BMC Recovery for DB2
- BMC Object Administration for DB2
- BMC Next Generation Technology Database Administration for DB2
- Database Administration for DB2

CHANGE MANAGER invokes the NGT Recover IMPORT command on the receiving subsystem during phase 2 of the migration to migrate all the image copy data to the receiving table space that is identified by the command. The IMPORT command uses the object identifiers found in the EXPORT command’s migration file and translates them to the object identifiers found in the receiving catalog.

Figure 51 on page 302 shows an example of the -BMCV command that is used with the IMPORT command.

### Figure 51: -BMCV command—IMPORT command

```
-BMCV 000700
IMPORT USING RDADQL.DEGA.WORKDXX.EX00001
CHANGE TABLE FROM RDADQL.TB0709A1 TO RDADQL2.TB0709A1
CHANGE TABLE FROM RDADQL.TB0709B1 TO RDADQL2.TB0709B1
CHANGE TABLE FROM RDADQL.TB0709C1 TO RDADQL2.TB0709C1
CHANGE TABLE FROM DQ2CLG2.T_T03PBGY TO DQ2CLG4.T_T03PBGY4
INDEXES NO
SYNC REPLACE
RESET
```
The IMPORT command uses the options listed in Table 21 on page 303. For more information, see BMC Next Generation Technology Recover for DB2 for z/OS Reference Manual.

### Table 21: IMPORT command options

<table>
<thead>
<tr>
<th>Option</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANGE TABLE FROM TO</td>
<td>Explicitly lists the tables that have a change to the creator or name</td>
</tr>
<tr>
<td>Option</td>
<td>Notes</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
</tbody>
</table>
| INDEXES | Specifies how NGT Recover should handle indexes using the IMPORT command:  
  - INDEXES AUTO specifies that NGT Recover recovers indexes if information is available for them in the migration data set. Any index not in the migration data set, but defined on the target system is rebuilt. This is the default value.  
  - INDEXES RECOVER specifies that indexes on the target system are recovered if they have entries in the migration data set.  
  - INDEXES REBUILD specifies that all indexes on the target system are rebuilt regardless of whether they have entries in the migration data set.  
  - INDEXES NO specifies that indexes on the target system are left unchanged. |
| SYNC | Specifies whether the migration file should include all spaces in the IMPORT command or only those spaces that have changed since the last migration  
  - REPLACE replaces all table spaces regardless of whether they have changed or not.  
  - AUTO replaces only table spaces that have changed since the last time those table spaces were imported. |
| RESET | Instructs NGT Recover to reset the recovery log points in each data page to 0. The log points must be reset when you are migrating data from one DB2 non-data-sharing subsystem to another subsystem because the log point values on the sending subsystem could be meaningless or misleading on the receiving subsystem. |
| OUTCOPY YES | Instructs NGT Recover to take an image copy of the table spaces during the processing of the IMPORT command. CHANGE MANAGER generates this option in the worklist when you select the UTILCOPY Analysis override option. The image copy data sets are dynamically allocated. |
| CLONE | Instructs NGT Recover to import clone table data |

After invoking the NGT Recover IMPORT command, CHANGE MANAGER invokes the NGT Recover REBUILD INDEX command to re-create the indexes and auxiliary indexes. If the table space is created implicitly, the character T precedes the command text. Execution looks up the table space name during processing and
substitutes it for the table name before invoking the utility (see Figure 52 on page 305).

**Figure 52: -BMCV command—T form of REBUILD INDEX**

```
-BMCV 000850
T REBUILD INDEX(ALL)
TABLESPACE RDADQL TB07090E1
AUX LOB
NOWORKDDN
REDEFINE NO
```

-**CHEK (IBM CHECK DATA or CHECK LOB utility)**

The -CHEK command invokes the IBM CHECK DATA or CHECK LOB utility on table spaces.

The -CHEK command can be processed in parallel. For more information about the worklist parallelism feature of the Database Administration and BMC Object Administration for DB2 solutions, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

**CHECK DATA utility**

The CHECK DATA utility identifies violations of referential and table check constraints in table spaces.

If a table space is created explicitly, the command is executed as shown in Figure 53 on page 305.

**Figure 53: -CHEK command—CHECK DATA utility**

```
-CHEK 001300
CHECK DATA TABLESPACE DQ2CL9.IXML
INCLUDE XML TABLESPACES ALL
ERRDDN SYSER001 WORKDDN SYSUT001
EXCEPTIONS 0
SORTDEV SYSDA
```

If the table space is created implicitly, the command text is preceded with the character T and the table space name is set to the table name. The Execution component looks up the table space name during processing and substitutes it for the table name before invoking the utility. This command might also be used due to referential integrity considerations.
Note
The keywords INCLUDE XML TABLESPACES ALL are added only if there are XML columns.

CHECK LOB utility

When the NGT Copy EXPORT command and NGT Recover IMPORT command are used to migrate an image copy that contains auxiliary table spaces, CHANGE MANAGER invokes the IBM CHECK LOB utility.

The CHECK LOB utility identifies inconsistencies in the auxiliary table spaces and invalid LOB values.

If the image copies that are used for the base table space and the auxiliary table space are inconsistent, the NGT Recover IMPORT command sets the base table space to a status of auxiliary check pending (ACHKP) and the auxiliary table space to a status of check pending (CHKP). CHANGE MANAGER first invokes the CHECK LOB utility for each auxiliary table space to change the status of the table spaces (see Figure 54 on page 306). Then, CHANGE MANAGER invokes the IBM CHECK DATA utility to change the status of the base table space.

Figure 54: -CHEK command—CHECK LOB utility

```
-CHEK 008450
  CHECK LOB TABLESPACE DQZ009.L02LOBI4
  EXCEPTIONS 0
  SORTDEVT SYSDA
-SYNC 008500 CHECK LOB FOR TS DQZ009.L02LOBI4 COMPLETE
-CHEK 008550
  LOBTBO DQZ009
  LOBTN_T_T03STMP
  LOBCOL CHARLOB
  CHECK LOB TABLESPACE LOB.TS
  CLONE
  EXCEPTIONS 0
  SORTDEVT SYSDA
-SYNC 008600 CHECK LOB FOR TS DQZ009.L03LOBI1 COMPLETE
-CHEK 008750
  CHECK LOB TABLESPACE DQZ009.L07LOBI1
  CLONE
  EXCEPTIONS 0
  SORTDEVT SYSDA
-SYNC 008800 CHECK LOB FOR TS DQZ009.L07LOBI1 COMPLETE
```

For more information, see the IBM documentation.
**-CMD (DB2 command)**

The -CMD command issues commands to DB2.

Execution assumes that the text field of the statement is a DB2 command and inserts a leading dash (-), if not supplied. Figure 55 on page 307 shows an example of the -CMD command.

**Figure 55: -CMD command**

```
-CMD 002500 STOP DATABASE(ASUETTA)
```

*Note*

The command text must be on the first line of the command.

The execution of certain utilities requires that a database or table space is in the stopped state. Execution uses this command to issue a DB2 STOP command on the object before continuing.

The Execution component then tests the status of the object to see whether the object has stopped. If the object has not stopped, Execution uses the value in the STOPWAIT parameter to determine how long to wait for the object to stop. If no STOPWAIT value exists, worklist processing stops immediately. The Execution component checks the object at two-minute intervals. The STOPWAIT value indicates the number of intervals. If the object does not stop, you can simply restart the execution job.

You can use -CMD to issue other DB2 commands and return the results to AEXPRINT. If any errors are found when a DB2 command is executing, the processing of the worklist stops.

**-COPY (IBM COPY utility)**

The -COPY command invokes the IBM COPY utility for table spaces that have been rebuilt or that have tables that are rebuilt or added.

This command and its parameters are passed to DB2.

You can run the COPY utility to perform the following functions:

- Dynamically allocate data sets
- Use generation data groups (GDGs)
Use of -COPY to dynamically allocate data sets

The utility can dynamically allocate the data sets that are used to create valid image copies.

JCL Generation uses the symbolic variable that is specified in the JCL Generation Data Set Options for Copies panels to create a TEMPLATE descriptor for Analysis. When generating the -COPY command, Analysis specifies a TEMPLATE descriptor for local and recovery primary and backup copies. The DSN parameter in the descriptor specifies the data set name for the object that is copied. Figure 56 on page 308 shows an example of a -COPY command that uses dynamic allocation.

**Figure 56: -COPY command—dynamic allocation of data sets**

```
-COPY 001650
  TEMPLATE DCPYLOCP
  DSN 'RDAMCG.&UT..A.&TS.'
  UNIT CART
  VOLCNT 99
  COPY TABLESPACE ACMX01.X02PSD
  COPYDDN (DCPYLOCP)
```

If you specify a threshold value for the size of the data set, Analysis specifies TEMPLATE descriptors for the primary data set and an alternate data set. If the threshold is exceeded, the utility can perform template switching and use the alternate data set. Figure 57 on page 308 shows an example of a -COPY command that uses two TEMPLATE descriptors.

**Figure 57: -COPY command—primary and alternate TEMPLATE descriptors**

```
-COPY 000400
  TEMPLATE SCPYLOCP
  DSN 'RDACRJ.&DB..&SN..LP000001'
  UNIT ALTUNIT
  DATACLAS ALTDC
  MGMTCLAS ALTMC
  STORCLAS ALTSC
  TEMPLATE DCPYLOCP
  DSN 'RDACRJ.&DB..&SN..LP000001'
  UNIT SYSDA
  LIMIT(10 CYL, SCPYLOCP)
```

**Note**

You cannot stack copy data sets that have been dynamically allocated on a tape.

Use of -COPY with generation data groups

When you use GDG data sets to store the image copies, a request for one copy per partition is treated like another image copy for that data set.

This action could result in the loss of archived data.
Notes for -COPY

For some objects, Analysis generates the -COPY command differently.

Note the following items when the -COPY command is used in the worklist:

■ If the table space is created explicitly, the command is executed as shown in Figure 58 on page 309.

Figure 58: -COPY command

COPY 000730
COPY TABLESPACE RESTATE.PROPERTY
DEVT TAPE COPYDDN SYSC1001

■ If the table space is created implicitly, the command text is preceded with the character T. Execution looks up the table space name during processing and substitutes it for the table name before invoking the utility.

■ If DB2 created XML or LOB objects implicitly, the -COPY command includes the following parameters:
  — XMLTBO or LOBTBO, which identifies the owner of the table
  — XMLETN or LOBTEBN, which identifies the name of the table
  — XMLCOL or LOBCOL, which identifies the name of the XML or LOB column

The COPY command syntax also includes the name of the XML or LOB table space that supports the XML or LOB column in the base table.

-ENDG (End global commands)

For the Database Administration, BMC Next Generation Technology Database Administration for DB2, and BMC Object Administration for DB2 solutions, the -ENDG command indicates the end of a group of global CHANGE MANAGER worklist commands.

The -ENDG command follows the -BEGG, -TIME, -GLID, -SSID, and -WKID commands, as shown in “-BEGG (Begin global commands)” on page 265. The statements between the -BEGG and -ENDG commands are sent to each initiator when a worklist is initially run and when it is restarted. For more information about the -BEGG command, see “-BEGG (Begin global commands)” on page 265.
-ENDP (End parallel processing)

For the Database Administration and BMC Object Administration for DB2 solutions, the -ENDP command indicates the end of a set of UOWs in the worklist that are processed in parallel.

The -ENDP command follows a -BEGP command and one or more UOWs, which are delineated by -BEGU and -ENDU commands. All of the UOWs within the -BEGP and -ENDP commands must complete processing before the next UOW outside of the -BEGP and -ENDP group can be processed. “-BEGP (Begin parallel processing)” on page 265 shows an example of the -ENDP command. For more information about the -BEGP command, see “-BEGP (Begin parallel processing)” on page 265.

-ENDU (End unit of work)

For the Database Administration and BMC Object Administration for DB2 solutions, the -ENDU command indicates the end of a UOW.

The -ENDU command follows a -BEGU command and a single command or a sequence of commands. “-BEGU (Begin unit of work)” on page 267 shows an example of the -ENDU command. For more information about the -BEGU command, see “-BEGU (Begin unit of work)” on page 267.

-ERR (Error)

The -ERR command indicates that a modification must be made to the worklist before it can be executed.

Instructional warning messages display at the beginning of the worklist. After the required information is supplied, the -ERR command must be deleted from the worklist. If you do not delete the command before you attempt to generate JCL to execute the worklist, the JCL Generation component will fail. Figure 59 on page 310 shows an example of a warning message and the -ERR command.

Figure 59: -ERR command

```
*** A MIGRATE WORKLIST FOR LOCATION ** LOCATION_2 ** ON SSID * **
*****************************************************************************
*BMC45402W STORED PROCEDURE(S) WITH TABLE LOCATOR VARIABLES ARE *
* CONTAINED IN THIS WORKLIST. SUPPLY THE TABLE NAME FOR *
* EACH LOCATOR AND DELETE THE -ERR COMMAND(S) BEFORE *
* EXECUTING THE WORKLIST. *
* STORED PROCEDURE ACMQA01.SPOP0011 *
*****************************************************************************
*** END OF WARNING MESSAGES - WARNINGS WERE ISSUED ***
*****************************************************************************
*** ORIGINATOR OF WORKLIST WAS WORKID RDAFJR.$MIG ON SSID DEAH ***
```
-GLID (Global authorization ID)

The -GLID command sets the global authorization ID (GLID) for worklist execution.

When this command is used, the -SETS and -AUTHreset commands (-SETS USER or -AUTH USER) set the current authorization ID to the GLID, instead of the authorization ID of the user who is executing the worklist. Figure 60 on page 311 shows an example of the -GLID command.

Figure 60: -GLID command

```
-GLID 000300 GLID0003 GLOBAL SETS AUTHID
```

The -GLID command allows a user to generate a worklist that executes under a particular authorization ID. You might use this feature when the authorization ID running the worklist does not have the necessary DB2 authorization for actions in the worklist. You can use the -GLID command to set the -GLID to a secondary authorization ID that has the necessary authorization.

The -GLID command can exit to an installation-supplied security exit. You can use this exit for additional authorization ID control. For information about the security exit and about using 8-byte authorization IDs, see the ALTER and CHANGE MANAGER for DB2 User Guide, Volume 1.

For information about the AUTHSW option, see “Authorizations used for authorization switching” on page 362. This option determines whether -AUTH, -SETS, or both commands are used for authorization ID switching.

-GOTO (Bypass command)

The -GOTO command is used in conjunction with the STARTOVER AEXIN keyword that allows worklist commands to be bypassed for execution until the specified sync number supplied with the-GOTO command is reached.

The -GOTO command allows STARTOVER execution of a worklist to begin processing from a point other than the beginning of the worklist of phase. Prior
execution of the worklist might have completed unsuccessfully and requires re-execution following some committed work.

If the worklist is for a receive-type work ID, the command must manually be added to PHASE-2 and be the first command following the first -SYNC command of PHASE-2. The only commands before the -GOTO are -TIME, -SSID, -WKID, -GLID (optional), -MIGR (receive-type work ID) and the first -SYNC command of the worklist or phase.

The STARTOVER AEXIN keyword disables auto-restart processing for a work ID and utilities. This allows worklist processing to start at the beginning of a worklist or phase of a worklist if a two-phase migrate worklist was generated. BMC recommends that you do not use this command often.

Two command sequence numbers are specified with the -GOTO command, as shown in Figure 61 on page 312:

- The first is the command sequence number of the -GOTO command that must be in ascending order with the prior and subsequent commands around -GOTO.

- The second is the command sequence number of the command the user wishes to be the start of this execution of the worklist.

**Figure 61: -GOTO command**

```
-WKID 000150 RDACRJ.REORG
-SYNC 000200 START OF EXECUTION SYNCPOINT
-GOTO 000250 000750
```

**Note**

Verify that the tasks from a prior worklist execution were committed. Execute the worklist following the last task that was committed.

**-ISMT (Is table empty)**

The results of the -ISMT command determine whether a table is still empty at Execution. Tables that are empty at Execution, such as many of the tables in an ERP system, are not unloaded and loaded.

Analysis generates the -ISMT command under the following conditions:

- If the NOUNLOADEMPTY parameter is specified and Analysis determines that a table is empty
  
  A table is empty when DB2 catalog statistics show that the cardinality of the table is 0 (zero). No UNLOAD or LOAD statements will be generated for the table.

- If the DEFINE parameter is set to NO for an index or a table space
Analysis does not generate the -ISMT command if one of the following conditions exists:

- A base table is incomplete.
  
  A base table can be incomplete under the following conditions:
  
  — if the unique index for a ROWID column is missing
  
  — if the auxiliary table space, table, or index is missing

- The DB2 catalog statistics show that the cardinality of the table is -1, which indicates that statistics do not exist.

Execution performs a SELECT COUNT * for each of the tables that are specified in the -ISMT commands. If a nonzero count is returned for any -ISMT statement, Execution terminates after the last -ISMT or -ISMX command in the worklist. This action ensures that data loss will not occur. Figure 62 on page 313 shows an example of the -ISMT command.

**Figure 62: -ISMT command**

-`ISMT 000250 JEFF.TBR5428B`

**-ISMX (Is LOB column empty)**

The results of the -ISMX command determine whether a large object (LOB) column in an auxiliary table is still empty at Execution. If the auxiliary table is empty, it can be dropped and rebuilt.

Analysis generates the -ISMX command in an alter-type worklist under the following conditions:

- For a nonpartitioned base table space, Analysis determines that an auxiliary table to be dropped is empty.
  
  A table is empty when DB2 catalog statistics show that the cardinality of the table is 0 (zero).

- For a partitioned base table space, Analysis determines that all of the auxiliary tables that reference a LOB column in the base table are empty.

Execution performs the following command for each of the tables that are specified in the -ISMX commands:

```
SELECT COUNT(*) WHERE LENGTH(LOBColumnName) = 0
```
If a nonzero count is returned for any -ISMX statement, Execution terminates after the last -ISMX or -ISMT command in the worklist. This action ensures that auxiliary objects can be dropped and re-created without having to first drop the base table. Figure 63 on page 314 shows an example of the -ISMX command.

**Figure 63: -ISMX command**

```bash
-ISMX 000250 RDACRJ.TBB22P3 CLOBCOL
```

If a column is defined as a distinct type (or UDT), the column is followed by the schema name of the UDT and the base type, as shown in Figure 64 on page 314.

**Figure 64: -ISMX command—column defined as UDT**

```bash
-ISMX 000250 RDACRJ.TBB22P3 CLOBCOL RDACRJ.CLOB
```

If DB2 catalog statistics show that the cardinality of the auxiliary table for the LOB column is greater than 0 (the LOB column is not empty) or -1 (statistics do not exist), Analysis:

- Generates commands in the worklist to drop and rebuild the base table
  The base table must be dropped to allow the product to make changes to the auxiliary objects that contain data.

- Does not generate the -ISMX command

**-JCLP (JCL Generation parameter)**

The -JCLP command is used by the JCL Generation function to build the JCL for executing the worklist.

This command does not perform any action during worklist execution. Instead, it creates JCL statements with the appropriate parameters.

The CM/PILOT component of CHANGE MANAGER also uses the -JCLP command in a CM/PILOT worklist, as shown in Figure 65 on page 314.

**Figure 65: -JCLP command in a CM/PILOT worklist**

```bash
-JCLP 001100 AUC0 DDNAME WORKL001
DSNAME 'RDAMAF.ACM.WKL(CHGDBC0M)'
-JCLP 001200 AUC0 DDNAME AFXJCL
DSNMAE 'RDAMAF.ACM.JCL(CHGDBC0M)'
```

**Use of -JCLP to identify SSIDs**

The -JCLP command refers to the DB2 subsystem ID (SSID) of the sending subsystem in a two-phase migration.
The -JCLP command, along with the &MSSID symbolic variable, is used to reference the SSID or the MSSID for the SYSR nnnn unload data set, as shown in Figure 66 on page 315.

Figure 66: -JCLP command—DB2 SSID

-JCLP 000700 MIGR MSSID DEAH

The -JCLP command is also used with the &TSSID symbolic variable to reference a receiving subsystem ID (TSSID) for a copy data set. When the name of the DB2 subsystem ID (SSID) is specified in the Target SSID field on the Edit WORKID Migration Options panel for a migrate-type work ID, the Analysis component generates a -JCLP worklist command with the TSSID parameter. The TSSID parameter specifies the name of the receiving (or target) SSID. JCL Generation uses the name of the receiving SSID to build the execution JCL. For example, if the command shown in Figure 67 on page 315 is included in the worklist, JCL Generation uses the DEBA SSID for the receiving subsystem.

Figure 67: -JCLP command—TSSID parameter

-JCLP 000700 MIGR TSSID DEBA

If the name of an SSID is not specified in the Target SSID field (for example, an * is used), the value of the &SSID symbolic variable is used as the value for the &TSSID variable.

Note

If the copy data set is dynamically allocated, Analysis substitutes the &SSID variable for the &TSSID variable in the COPY OUTPUT descriptor. The copy utility resolves the &SSID variable.

Use of -JCLP to create partition-level image copies

Analysis can generate partition-level image copies of a partitioned table space or index in a database when utilities other than a copy utility are used.

JCL Generation uses different parameters with the -JCLP command, depending on the utility that is used and the number of partitions that are copied.

- If the REORG PLUS utility is used to reorganize a subset of partitioned table spaces or indexes in a database and to copy only those partitions that are being reorganized, JCL Generation uses the COPYPARTS parameter (see Figure 68 on page 315).

Figure 68: -JCLP command—COPYPARTS parameter

-JCLP 000650 BMCR COPYPARTS
  -BMCR 000700
  REORG TABLESPACE C7PCB1.TS35P5
If the LOADPLUS, REORG PLUS, or NGT Recover utility is used to create partition-level image copies of all of the partitions, JCL Generation uses the NUMPARTS parameter (see Figure 69 on page 316).

**Figure 69: -JCLP command—NUMPARTS parameter**

```
-JCLP 000600 BMCR NUMPARTS 5
  -BMCR 000650
    REORG TABLESPACE C7PCB1.TS35P5
    DELETEFILES NO
    UNLDDN SYSR1001
    WORKDDN (WRK1001)
    COPY YES
    REGISTER C001
    COPYDDN (C001)
    SORTDATA LOG NO
```

**Use of -JCLP for multiple work data sets**

If the LOADPLUS or REORG PLUS utilities use multiple SYSUT temporary work data sets to build nonclustering indexes for a table, Analysis inserts the -JCLP command before each -BMCL or -BMCR command in the worklist.

The -JCLP command specifies the LOADPLUS or REORG PLUS command and the maximum number of SYSUT data sets that can be specified with the NUMIX parameter.

For REORG PLUS, if the number of nonclustering indexes in a table is equivalent to or less than the maximum number of SYSUT data sets, JCL Generation creates one SYSUT data set for each nonclustering index. LOADPLUS functions in the same manner, if the number of all of the indexes in a table is equivalent to or less than the maximum number of SYSUT data sets. Figure 70 on page 316 shows an example of a -JCLP command that specifies that a maximum of two SYSUT data sets are used to reorganize a partitioned table space.

**Figure 70: -JCLP command—NUMIX parameter**

```
-JCLP 001100 BMCR NUMIX   2
  -BMCR 001150
    REORG TABLESPACE C7UTE1.TS39P254
    DELETEFILES NO
    UNLDDN SYSR1003
    WORKDDN (SUT)
    PART 1
    SORTDATA LOG NO
```
Use of -JCLP to multitask the unloading and loading of data

You can use UNLOAD PLUS and LOADPLUS to multitask the unloading and loading of data.

When multitasking occurs, JCL Generation uses the IPPARTS parameter in the -JCLP command to specify the number of partitions in a table space. Figure 71 on page 317 shows an example of the -JCLP command with the IPPARTS parameter.

Figure 71: -JCLP command—IPPARTS parameter

```
-JCLP 000550 BMCD IPPARTS 3
-BMCD 000600
  UNLOAD UNLOADDN (R1003)
  ACTIVE (YES,NO)
  DISCARDS 1
  NULLTYPE T1 NULLCHAR ?
  SELECT * FROM C7PCA1C.TB33A ORDER BY COLUMN_1
;```

Use of -JCLP to determine the sizes of data sets

If the LOADPLUS utility is used to load data from a table-controlled partitioned table, Analysis inserts the -JCLP command before the -BMCL command in the worklist.

The -JCLP command specifies the size of the index keys in the KEYLEN parameter. JCL Generation uses the key length when calculating the sizes of the SORTOUT and SYSUT data sets. Figure 72 on page 317 shows an example of the -JCLP command with the KEYLEN parameter.

Figure 72: -JCLP command—KEYLEN parameter

```
-JCLP 000650 BMCL KEYLEN 61
-BMCL 000700
  LOAD DATA REPLACE INDDN SYSR1001 ERRDDN SYSER001 DISCARDS 0 DISCARDDN SYSD1001 DELETENull NO EBCDIC CCSID(37,0,0) WORKDDN SYSUT001 LOADDN SORT0001 UNIQUEINTO YES INTO TABLE RDACRJ2.TB5 (COLUMN_1 POSITION(*) INTEGER, COLUMN_3 POSITION(*) SMALLINT, COLUMN_2 POSITION(*) CHAR(12), COLUMN_5 POSITION(*) DECIMAL(11,2), COLUMN_4 POSITION(*) CHAR(7), COLUMN_6 POSITION(*) DATE EXTERNAL) ;```

Chapter 5  Worklist commands    317
**-LCMD (List command)**

For the Database Administration, BMC Next Generation Technology Database Administration for DB2, and BMC Object Administration for DB2 solution, the -LCMD command executes SQL that lists the LOB (auxiliary) table spaces that support the LOB column in the base table.

DB2 creates these objects implicitly for a partition-by-growth table space. The -LCMD command includes the following parameters:

- LOBTBO, which identifies the owner of the table
- LOBTBN, which identifies the name of the table
- LOBCOL, which identifies the name of the LOB column

Analysis issues the -LCMD command after the -BMCL command invokes the LOADPLUS utility to load all of the columns in the base table, except the LOB columns. The -LCMD command executes a START DATABASE ACCESS(FORCE) command on each LOB table space in the list (see Figure 73 on page 318). Analysis uses the -LSQL command to turn logging off for each LOB table space before it uses the -LOLB command to load data in the LOB columns.

**Figure 73: -LCMD command**

```sql
- BMCL 001300
  LOAD DATA RESUME YES
  INDSN('RDACRJ2.MIGRATE.SYSREC.SR000001')
  ERRDNO SYSER001
  PRELOAD CONTINUE
  DISCARDS 1
  DISCARDN SYSDS001
  DELETESFILES NO
  ORDER YES
  EBCDIC CCSID(37,0,0)
  WORKDNO SUT
  LOADDN SORT0001
  UNIQUEINTO YES
  INTO TABLE MGPBG21.TX12GS
  (COL1_VCHAR
   POSITION(*) VARCHAR,
   COL2_SMALL
   POSITION(*) SMALLINT,
   COL3_ROWID
   POSITION(*) ROWID)

- LCMD 001400
  LOBTBO MGPBG21
  LOBTBN TX12GS
  LOBCOL COL4_CLOB
  START DATABASE (LOB ) SPACENAM (TS )
  ACCESS(FORCE)

- LSQL 001500
  LOBTBO MGPBG21
  LOBTBN TX12GS
  LOBCOL COL4_CLOB
```
-LDXT (IBM LOAD Extended Text utility)

The -LDXT command invokes the IBM LOAD utility.

Analysis uses the IBM LOAD utility when all of the following conditions exist:

- You are using multitasking to load tables in a partitioned table space.
- The LOAD command might contain a large number of columns and INTO TABLE PART clauses.

The LOAD utility loads the data that the -UNLI unload command created.

Figure 74 on page 319 shows an example of the -LDXT command.

Figure 74: -LDXT command

-LDXT 001400
  TEMPLATE SYRC
    DSN 'RDAJZB4.DBDC.JDC1102F.SR000004.P&PART.'
    UNIT SYSDA
  TEMPLATE SYDS
    DSN 'RDAJZB4.JDC1102F.SD000004.P&PART.'
    UNIT SYSDA
    SPACE(1,1) TRK
  LOAD DATA REPLACE
  ERRDDN SYSER001
  DISCARDS 1
  UNICODE CCSID(367,1208,1200)
  WORKDDN SYSUT001
  ENFORCE NO
  REUSE
  INTO TABLE J9U8LJ2C.T_T08TCP
  IGNOREFIELDS YES
  PART 1 INDON SYRC
  DISCARGDN SYDS

(,
  COLC_0
    POSITION(*) CHAR(20),
  COLC_1
    POSITION(*) CHAR MIXED (4),
  COLI_2
    POSITION(*) INTEGER,
  COLBI_3
    POSITION(*) BIGINT,)
ALTER and CHANGE MANAGER worklist commands

COLBIN_4
  POSITION(*) BINARY,
COLVB_5
  POSITION(*) VARBINARY,
COLCI_5B
  POSITION(*) VARCHAR CLOBF.
COLVC_6
  POSITION(*) VARCHAR MIXED.
COLT_7
  POSITION(*) TIME EXTERNAL,
COLVC_9
  POSITION(*) VARCHAR MIXED,
COLRI_5A
  POSITION(*) ROWID)
INTO TABLE J9U8LJ2C.T_T08TCP
  IGNOREFIELDS YES
  PART 2 INDDN SYRC
  DISCARDDN SYDS
(
  COLC_0
    POSITION(*) CHAR(20),
  COLC_1
    POSITION(*) CHAR MIXED (4),
  COLI_2
    POSITION(*) INTEGER,
  COLBI_3
    POSITION(*) BIGINT.
  COLBIN_4
    POSITION(*) BINARY,
  COLVB_5
    POSITION(*) VARBINARY,
COLCI_5B
    POSITION(*) VARCHAR CLOBF.
COLVC_6
    POSITION(*) VARCHAR MIXED.
COLT_7
    POSITION(*) TIME EXTERNAL,
COLVC_9
    POSITION(*) VARCHAR MIXED,
COLRI_5A
    POSITION(*) ROWID)
INTO TABLE J9U8LJ2C.T_T08TCP
  IGNOREFIELDS YES
  PART 3 INDDN SYRC
  DISCARDDN SYDS
(
  COLC_0
    POSITION(*) CHAR(20),
  COLC_1
    POSITION(*) CHAR MIXED (4),
  COLI_2
    POSITION(*) INTEGER,
  COLBI_3
    POSITION(*) BIGINT.
  COLBIN_4
    POSITION(*) BINARY,
  COLVB_5
    POSITION(*) VARBINARY,
-LOAD (IBM LOAD utility)

The -LOAD command invokes the IBM LOAD utility to load the data that the -BMCD or -UNLI unload command created.

The load statement includes the unload data set identifier that is created for the unload command. This command and its parameters are passed through to DB2. The definitions are ordered by the new column sequence and refer to the new column names. New columns with NULL=YES or DEFAULT=YES are omitted.

You can run the LOAD utility to perform the following functions:

- Load data in XML and LOB columns
- Migrate tables with ROWID columns
- Migrate only data

Figure 75 on page 321 shows an example of the -LOAD command with the REPLACE option.

Figure 75: -LOAD command with REPLACE option

```sql
LOAD 020000 LOAD DATA REPLACE INDDN SYSR1002 ERRDDN SYSER001
```
After a LOAD RESUME fails, an SQL DELETE statement executes before the LOAD restarts. The SQL statement cleans up any data that is left in the table by the failed LOAD attempt. Figure 76 on page 322 shows an example of the -LOAD command with the RESUME YES option.

**Figure 76: -LOAD command with RESUME YES option**

```
-LOAD 014000 DELETE FROM
LOAD DATA  INDDN SYSR1012 RESUME YES
ERRDDN SYSER001
DISCARDS 0
DISCARDDN SYSD1012
WORKDDN SYSUT001
LOG NO
INTO TABLE V6MOVIE.SRTBT1
(ACTION           POSITION(*) CHAR(1),
SALARY            POSITION(*) DECIMAL,
COUNT             POSITION(*) INTEGER,
COL_VAR           POSITION(*) VARCHAR)
```

When you create a data-only migrate worklist, the Analysis component generates and groups all of the -SQL DELETE commands before the first LOAD command. These commands appear in phase 2 of the worklist, which is indicated by the -MIGR PHASE-2 command.

### Use of -LOAD to load data in LOB and XML columns

The IBM LOAD utility can be used to load data contained in LOB and XML columns that the IBM UNLOAD utility unloaded.

The data set that the SYRC TEMPLATE descriptor references contains all of the data and a reference to a file name for each LOB or XML column in the table. Figure 77 on page 322 shows an example of a -LOAD command that loads XML data.

**Figure 77: -LOAD command—loading XML data**

```
-LOAD 001350 DELETE FROM
MGX02NG.T27N01LONG
TEMPLATE SYRC
DSN 'RDACRJ.MXML.SR000001'
UNIT SYSDA
LOAD DATA  RESUME YES
INDDN SYRC
ERRDDN SYSER001
```
Analysis generates the IGNOREFIELDS YES parameter for the LOAD statement. This parameter allows the NULL nnn column to be generated, but not referenced. The column list does not include the DB_GENERATED_DOCID_FOR_XML column.

**Use of -LOAD to migrate tables with ROWID columns**

You can modify or migrate tables that contain ROWID columns with data.

If the ROWID column is defined as GENERATED ALWAYS, ALTER and CHANGE MANAGER omit the ROWID column in the UNLOAD and LOAD statements. If the ROWID column is defined as GENERATED BY DEFAULT, ALTER and CHANGE MANAGER include the ROWID column in the UNLOAD and LOAD statements. The column is unloaded last in the sequence of columns and is also loaded last, even though it might not be the last column in the table. When a table that contains a ROWID column is unloaded, Analysis lists all of the columns that are unloaded in the UNLOAD statement in the worklist.
Use of -LOAD to migrate only data with FORCELOADREPLACE

When you specify the FORCELOADREPLACE keyword in the ALUIN input stream, Analysis generates one of a few forms of the LOAD DATA syntax.

Analysis generates the following forms of the syntax:

- **LOAD DATA RESUME YES INTO TABLE**

  Analysis uses this syntax (Figure 78 on page 324) when you are migrating only data from:

  - A single table at the table level of a segmented table space
  - The second and succeeding tables of multiple tables at the table space level of a segmented table space

Analysis does not assume that the structures on the sending and receiving subsystems are identical. The load utility deletes the existing data in the table before loading.

**Figure 78: -LOAD command—LOAD DATA RESUME YES INTO TABLE**

```sql
-LOAD 000600 DELETE FROM J1ALLN1.T_T03MSEG_A
TEMPLATE SYRC
  DSN 'RDAJZB4.DEGA.JB111288.SR000002'
  UNIT SYSDA
  LOAD DATA RESUME YES
  INDDN SYRC
  ERRDDN SYSE001
  DISCARDS 1
  DISCARDDN SYSDS001
  EBCDIC CCSID(37,0,0)
  WORKDDN SYSUT001
  LOG NO
  ENFORCE NO
  INTO TABLE J1ALLN1.T_T03MSEG_A
  IGNOREFIELDS YES
  (.Null001 POSITION(*) CHAR(1),
   NULL002 POSITION(*) CHAR(20)
   NULL003 POSITION(*) BIGINT
   NULL004 POSITION(*) BINARY
   NULL005 POSITION(*) CHAR(1),
   COLC_0 NULLIF NULL001=X'FF',
   COLBI_1 NULLIF NULL002=X'FF',
   COLBN_2 NULLIF NULL003=X'FF',
   COLBN_3 NULLIF NULL004=X'FF',
   NULL001
   NULL002
   NULL003
   NULL004
   NULL005
   COLC_0 POSITION(*) CHAR(20)
   COLBI_1 POSITION(*) BIGINT
   COLBN_2 POSITION(*) BINARY
   COLBN_3 POSITION(*) BINARY
   COLC_0
   COLBI_1
   COLBN_2
   COLBN_3)
```

ALTER and CHANGE MANAGER worklist commands
LOAD DATA REPLACE INTO TABLE

Analysis uses this syntax (Figure 79 on page 325) when you are migrating only data from:

— A single table at the table space level of a segmented table space

— The first table of multiple tables at the table space level of a segmented table space

— An entire table space at the table space level of a partitioned table space

Analysis assumes that the structures on the sending and receiving subsystems are identical, and generates the LOAD statements based on the structure of the receiving subsystem. The load utility replaces the data in the table space.

Figure 79: -LOAD command—LOAD DATA REPLACE INTO TABLE
Notes for `-LOAD`

For some objects, Analysis generates the `-LOAD` command differently.

Note the following items when the `-LOAD` command is used in a worklist:

- A SYSMAP data set is always generated by JCL Generation if the worklist contains `-LOAD`.
- If a table space is being created in the worklist, Analysis includes the REUSE parameter in the `-LOAD` command.
If you are using a tape data set for SYSUT with IBM LOAD, you must manually edit the worklist and JCL to ensure that multiple instances of IBM LOAD use unique work data sets and ddname for SYSUT.

For more information, see the IBM documentation.

-MIGR (Migrate phase)

The -MIGR command identifies the migration phases.

Phase 1 contains the statements that are executed on the sending subsystem. Phase 2 contains the statements that are executed on the receiving subsystem(s). In the case of a single-phase worklist for migrating within the same subsystem, only the -MIGR phase 1 command is necessary. Figure 80 on page 327 shows an example of the -MIGR phase 1 command.

Figure 80: -MIGR phase 1 command

-MIGR 000100 PHASE-1 START OF SECTION TO BE RUN ON SENDING SYSTEM

Figure 81 on page 327 shows an example of the -MIGR phase 2 command.

Figure 81: -MIGR phase 2 command

-MIGR 001700 PHASE-2 START OF SECTION TO BE RUN ON RECEIVING SYSTEM

-NGTU (NGT utilities)

The -NGTU command invokes NGT utilities.

You can run the NGT Reorg utility to:

- Perform SHRLEVEL CHANGE data reorganizations
- Create partition-level image copies

In a worklist, NGT Reorg always uses dynamic allocation. You can specify NGT Reorg to dynamically allocate the data sets either by:

- Using the OUTPUT JCL statement to use the definitions from your product options file (POF)
- Using the NGTAUTO JCL statement to use your configured NGT automation control points
Example of reorganization with an OUTPUT parameter

To use the OUTPUT parameter, you must set the USE_NGT_AUTO POF keyword to N.

In a worklist, you can output to the following types of data sets with NGT Reorg by using the OUTPUT parameter:

- Copy—COPYDDN, RECOVERYDDN
- Discard—DISCARDDDN
- Punch—PUNCHDDN

**Example**

```
-NGTU 000450
  OUTPUT DCPYLOCP
  UNIT SYSDA
  DSNAME RDADQL.DQG01220.TSSEGBP1.LP0001

  REORG TABLESPACE DQBPCH02.TSSEGBP1
      COPYDDN (DCPYLOCP)
```

Examples of reorganization without an OUTPUT parameter

To omit the OUTPUT parameter and use your configured NGT automation control points, you must set the USE_NGT_AUTO POF keyword to Y.

You can omit the OUTPUT syntax and corresponding DDNs from the -NGTU command in the worklist. In that case, the NGT utilities’ embedded automation retains full control of all data set names and attributes.

**Example**

To reorganize individual partitions of an index:

```
-NGTU 000400
  REORG INDEX DQREOR02.P_A_ACNTBLK_MBR VA_A03
      PART 5

-NGTU 000400
  REORG INDEX DQREOR02.P_A_ACNTBLK_MBR VA_A03
      PART 8
```

To reorganize a table space that has no LOBS:

```
-NGTU 000650
  REORG TABLESPACE DQREOR02.TSICPA03
```
-RBLD (IBM REBUILD INDEX utility)

The -RBLD command invokes the IBM REBUILD INDEX utility.

Use -RBLD followed by the parameters for the REBUILD command. This command and its parameters are passed to DB2. Figure 82 on page 329 shows an example of the -RBLD command.

**Figure 82: -RBLD command**

```
-RBLD 000350
  REBUILD INDEX (RDAJZB.IXSS1AN0)
  REUSE
  SORTKEYS
  SORTDEVT SYSDA
  STATISTICS
  KEYCARD
  HISTORY ALL
```

**Note**

If the PIECESIZE attribute is altered and an index is rebuilt, the REUSE parameter is not used in the -RBLD command.

If a cloned table exists for a base table, and an index is being created for the base table, new indexes on tables with clones are marked DEFER YES in the create statement, and the CLONE keyword is added to the -RBLD REBUILD INDEX statement, as shown in Figure 83 on page 329.

**Figure 83: -RBLD INDEX CLONE command**

```
-RBLD 000450
  REBUILD INDEX (RJCLONE2.I_T02PBG)
  CLONE
  REUSE
  SORTKEYS
  SORTDEVT SYSDA
```

For more information, see the IBM documentation.

-REBD (DB2 REBIND)

The -REBD command invokes the DB2 REBIND command and is used for all application plans and packages that are invalidated by changes to other objects.
Authorization ID switching ensures that the rebind applies to the original creator of the plan or package. The parameters that are used are from the DB2 catalog. Figure 84 on page 330 shows an example of the -REBD command for plans.

**Figure 84: -REBD command for plans**

```
-AUTH 000690 JEO
-REBD 000700 REBIND PLAN ALUAPL1
   VALIDATE(BIND) ISOLATION(CS)
   ACQUIRE(USE)  RELEASE(COMMIT) EXPLAIN(NO)
```

Figure 85 on page 330 shows an example of the -REBD command for packages.

**Figure 85: -REBD command for packages**

```
-AUTH 000690 JEO
-REBD 000700 REBIND PACKAGE (ACM721_D_MAIN.AEXARNAM
   OWNER(RDAMLB) QUALIFIER(ACM721SN)
   VALIDATE(BIND) ISOLATION(CS)
   RELEASE(COMMIT) EXPLAIN(NO)
```

**-REOR (IBM REORG utility)**

The -REOR command invokes the IBM REORG utility.

This command and its parameters are passed to DB2. Figure 86 on page 330 shows an example of the -REOR command.

**Figure 86: -REOR command**

```
-REOR 000700
   REORG INDEX DEMOCJ.ASCIIN3
   SHRLEVEL CHANGE
   MAXRO 0
   WORKDDN (WRK1001)
   STATISTICS
      KEYCARD
      HISTORY ALL
```

Table spaces that are VCAT defined are reorganized in two phases. The RESTART parameter is used in the -REOR command in the second phase of the reorganization, as shown in Figure 87 on page 330.

**Figure 87: -REOR command—two-phase reorg**

```
-REOR 000250
   REORG TABLESPACE ACMX02A.X20NV
   UNLOAD PAUSE
   UNLDDN SYR1001
   SORTKEYS
   SORTDATA LOG NO

... 
-REOR 000650 .RESTART
   REORG TABLESPACE ACMX02A.X20NV
   UNLOAD PAUSE
```
In some cases, the only changes that you make in a work ID can be accomplished by using ALTER statements and online reorganizations (for example, online schema changes). In those cases, you can use IBM REORG to perform an online reorg (SHRLEVEL CHANGE) of your table spaces. Figure 88 on page 331 shows an example of the parameters that the utility uses.

**Figure 88: -REOR command—online reorg**

```
REORG TABLESPACE DEMSMP01.TS003PV
UNLDDN SYR1001
COPYDDN (SYCL0001)
SORTKEYS
SORTDATA LOG NO
SHRLEVEL CHANGE
MAXRO 0
MAPPINGTABLE RDACRJ.CRJTBMAP
```

ALTER and CHANGE MANAGER support the dynamic allocation of data sets for IBM REORG. When the Analysis component generates the worklist, the component includes the names of the data sets. You can enable or disable dynamic allocation by using the DYNREORG installation option, the DYNREORG or NODYNREORG ALUIN keywords, or an option on the Analysis Utility Dataset Options panel. Figure 89 on page 331 shows an example of the syntax that IBM REORG uses for the data sets.

**Figure 89: -REOR command—dynamically allocated data sets**

```
REOR DD0550
TEMPLATE SYRC
   DSN 'RDACRJ.DEGA.RB10206D.SR000001'
   UNIT SYSDA

TEMPLATE SYDS
   DSN 'RDACRJ.RB10206D.SD000001'
   UNIT SYSDA

TEMPLATE SYPU
   DSN 'RDACRJ.RB10206D.STEP1.SP000001'
   UNIT SYSDA

TEMPLATE DCPYLOCP
   DSN 'RDACRJ.RB10206D.T02ASSEG.P000.LP000001'
   UNIT SYSDA

   REORG TABLESPACE CJBASIC.T02ASSEG
   UNLDDN SYRC
   COPYDDN (DCPYLOCP)
   SORTKEYS
   SORTNUM 32
   SORTDEVT SYSDA
   SORTDATA LOG NO
   SHRLEVEL REFERENCE
```

For more information, see the IBM documentation.
**Note**
The utility command starts on the second line of the statement. The first line is reserved for parameters that are passed to the utility.

---

**-REXC (REXX and CLIST macro execution)**

The -REXC command allows a REXX or CLIST subroutine to execute outside of the worklist and return back to the worklist after the subroutine is completed.

In order to use this feature, you must perform the following actions:

- Allocate any files required for the EXEC's processing either dynamically or via JCL.

  - For implicit data set location of the REXX exec the worklist must contain:

    ```
    -REXC 000450 MYREXX
    ```

    where `MYREXX` is the member containing the REXX exec. The user must manually insert in the Execution JCL, for foreground or batch processing, the following DD statement:

    ```
    //SYSPROC DD DSN='loc.of.rexx.exec'
    ```

    Alternatively, you can specify the name of a partitioned data set in which a REXX EXEC is a member for the Sysexec option on the JCL Generation Jobcard Options panel. The name of the data set is included in the //SYSEXEC DD in the Execution JCL, as shown in the following example:

    ```
    //SYSEXEC DD DSN='loc.of.rexx.exec'
    ```

  - For explicit data set location of the REXX exec the worklist must contain:

    ```
    -REXC 000450 EX 'loc.of.rexx.exec(MYREXX)'
    ```

- Ensure that the return code for the REXX exec is sent back through IKJEFT1B. This means that a final REXX statement of `EXIT RC` (where `RC` is the return code to pass back to AEXEMAIN) must be included.

---

**Note**
If the required return code is not passed back to AEXEMAIN, unless a TSO command abends, AEXEMAIN will receive a return code zero (RC=0) and continue to process the worklist.
WARNING

BMC strongly recommends that you evaluate any REXX exec inserted in worklists in a test environment before implementing production versions of the macro. Particular care should be taken to determine whether any data sets are freed by the REXX exec that are allocated by ALTER or CHANGE MANAGER. Execution of REXX macros that free data sets allocated by ALTER or CHANGE MANAGER could have drastic and unpredictable results.

-RNAM (Rename object)

The -RNAM command is generated whenever a DB2 object (or table column or view column) is renamed.

The -RNAM command records the changed name in the CHANGE MANAGER database. The Compare component of CHANGE MANAGER uses these name changes to resolve renamed objects (so that CDL can be generated properly) and to prevent the loss of data when a baseline is one of the inputs for the comparison.

The RENAME TABLE SQL command renames a table, relevant check constraints, and foreign keys. CHANGE MANAGER accounts for creating and dropping views, synonyms, and aliases and rebinding plans and packages.

For example, if you establish a baseline on a table, then rename a column in the table, and then recover back to the baseline, Compare can generate CDL to successfully restore the old column name. Otherwise, all of the data in that column might be lost when the table is restored.

The -RNAM command must be followed by the object type, the old name (up to three parts, separated by periods), and the new name. In Figure 90 on page 333, CO indicates a table column and the column LNAME is renamed to LAST.

Figure 90: -RNAM command

-RNAM 000100 CO GEC.NAMES.LNAME LAST

-RNST (IBM RUNSTATS utility)

The -RNST command invokes the IBM RUNSTATS utility on table spaces or indexes.

The -RNST command can be processed in parallel. For more information about the worklist parallelism feature of the Database Administration and BMC Object Administration for DB2 solutions, see the ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2.
Note the following items when the -RNST command is used in a worklist:

- The utility command starts on the second line of the statement. The first line is reserved for parameters that are passed to the utility.

- If a table space has been rebuilt or has tables that are rebuilt or added, Analysis creates a -RNST command for the table space. If the table space is created explicitly, the command is executed as shown in Figure 91 on page 334.

  **Figure 91: -RNST command**

  ```
  -RNST 004450
  RUNSTATS TABLESPACE KIF8D22A.MAF8S22E
  HISTORY NONE
  INDEX (ALL)
  KEYCARD
  ```

- If the table space is created implicitly, the command text is preceded with the character T. Execution looks up the table space name during processing and substitutes it for the table name **before invoking the utility**. This command might also be used due to referential integrity considerations.

- If the DB2 catalog history tables will be updated with all of the statistics that have been collected, Analysis adds the HISTORY ALL parameter to the command. Conversely, if the history tables will not be updated, Analysis adds the HISTORY NONE parameter to the command.

- If an index is created or rebuilt for a table space that had no table changes, Analysis creates a -RNST command for the index.

- If DB2 created XML or LOB objects implicitly, the -RNST command includes the following parameters:
  - XMLTBO or LOBTBO, which identifies the owner of the table
  - XMLTBN or LOBTBN, which identifies the name of the table
  - XMLCOL or LOBCOL, which identifies the name of the XML or LOB column

  The RUNSTATS command syntax also includes the name of the XML or LOB table space that supports the XML or LOB column in the base table.

  For more information, see the IBM documentation.

---

**-SETA (SET SCHEMA statement)**

The -SETA command sets the schema creator for SQL statements.

The command also records the schema creator when the Execution component restarts a worklist. After the object is created, the -SETA command sets the schema
creator to the default for subsequent -SQL commands. Figure 92 on page 335 shows an example of the -SETA command.

**Figure 92: -SETA command**

```
-SETS 000550 SET CURRENT SQLID = 'J1OSS9A'
-SETA 000600 SET SCHEMA = 'J1CSS9A'
-SQL 000650 CREATE TABLE T_T03MSEG_B
   ( COLSI_2 SMALLINT NOT NULL WITH DEFAULT
   , COLIN_1 INTEGER NOT NULL WITH DEFAULT 1
   , COLIN_3 INTEGER NOT NULL WITH DEFAULT
   , COLDT_9 DATE NOT NULL WITH DEFAULT '1961-05-06'
   , COLTS_11 TIMESTAMP NOT NULL GENERATED BY DEFAULT
      FOR EACH ROW ON UPDATE AS ROW CHANGE TIMESTAMP
   )
   CCSID EBCDIC
   IN J1CSS9A.T03MSEG
-SQL 000700 CREATE TABLE T_T03MSEG_A
   ( COLSI_2 SMALLINT NOT NULL WITH DEFAULT
   , COLIN_1 INTEGER NOT NULL WITH DEFAULT 1
   , COLIN_3 INTEGER NOT NULL WITH DEFAULT
   , COLDT_9 DATE NOT NULL WITH DEFAULT '1961-05-06'
   , COLTS_11 TIMESTAMP NOT NULL GENERATED BY DEFAULT
      FOR EACH ROW ON UPDATE AS ROW CHANGE TIMESTAMP
   )
   CCSID EBCDIC
   IN J1CSS9A.T03MSEG
-SETA 000750 SET SCHEMA = DEFAULT
-SETS 000800 SET CURRENT SQLID = USER
-SYNC 000850 END OF TABLE SECTION
```

**-SETP (SET CURRENT PATH statement)**

The -SETP command is used to set the current path before creation of objects such as triggers and views.
The current path is reset to system path after the objects are created. Figure 93 on page 336 shows an example of the -SETP command.

**Figure 93: -SETP command for system path**

```
-SETP 002700 SET CURRENT PATH = SYSTEM PATH
```

Figure 94 on page 336 shows an example of the -SETP command setting the current path to the system path and an additional schema.

**Figure 94: -SETP command with additional schema**

```
-SETP 002850 SET CURRENT PATH = "SYSIBM","SYSFUN","SYSPROC","RDAJFL2"
```

**Note**

Use the -SETP command instead of the -SQL command to set the path so that restart processing will work correctly.

---

**-SETS (Set current SQL ID)**

The -SETS command sets the current SQLID for all subsequent SQL statements.

The command also records the schema creator when the Execution component restarts a worklist. Figure 95 on page 336 shows an example of the -SETS command.

**Figure 95: -SETS command**

```
-SETS 000120 SET CURRENT SQLID = 'BSR1'
```

The products use the -SETS and -SETA worklist commands as follows (Figure 96 on page 336) when the following conditions exist:

- You create an unqualified alias, table, or index with a one-part name.
- The owner and the creator of the object are different.

**Figure 96: Commands for unqualified aliases, tables, or indexes**

```
-SETS 000900 SET CURRENT SQLID = 'SQLID'
-SETA 000950 SET SCHEMA = 'creator'
-SETA 001000 SET SCHEMA = DEFAULT
-SETS 001100 SET CURRENT SQLID = USER
```
When you create an unqualified view, trigger, or MQT, the products use the -SETS, -SETA, and -SETP commands (Figure 97 on page 337).

**Figure 97: Commands for unqualified views, triggers, or MQTs**

```
-SETS 001650 SET CURRENT SQLID = 'SQLID'
-SETA 001700 SET SCHEMA = 'creator'
-SETP 001750 SET CURRENT PATH = "SYSIBM","SYSFUN","SYSPROC","J10S59A"
-SQL 001800 CREATE
object onePartName
-SETP 001850 SET SCHEMA = DEFAULT
-SETS 001900 SET CURRENT SQLID = USER
-SETP 001950 SET CURRENT PATH = SYSTEM PATH
```

Note the following items when the -SETS command is used in a worklist:

- If the worklist uses a global authorization ID (GLID), a -SETS command with a value of USER sets the current SQLID to the GLID. If the worklist does not use a GLID, a -SETS command with a value of USER sets the current SQLID to that of the user who is executing the worklist.

- If the Analysis component cannot determine the qualifier of a table that is used in a view, a -SETS command is specified with a value of UNKNOWN.

- The -SETS command can exit to an installation-supplied security exit. You can use this exit for additional authorization ID control. For information about the security exit and about using 8-byte authorization IDs, see the ALTER and CHANGE MANAGER for DB2 User Guide, Volume 1.

- Use the -SETS command instead of an SQL command for setting the current SQLID. Restart processing might not set the SQLID correctly if an SQL command is used.

- For information about the AUTHSW option, see “Authorizations used for authorization switching” on page 362. This option determines whether -AUTH, -SETS, or both commands are used for authorization ID switching.

**-SPAC (Space estimation source)**

The -SPAC command identifies the existing object on which to base space estimation.

The -SPAC command is used for the following commands:

- -BMCC (NGT Copy)
- -BMCL (LOADPLUS)
-BMCR (REORG PLUS)

- BMCV (NGT Recover)

-COPY (IBM COPY)

For example, if a table space is dropped and then created with a new name, JCL Generation uses the -SPAC command to identify the original name of the object, since no statistics or physical data sets exist. When the NGT Copy, LOADPLUS, REORG PLUS, or NGT Recover utility is used to create a partition-level image copy of a partitioned table space or index, the -SPAC command is used to size the table space or index partition.

Figure 98 on page 338 shows an example of the -SPAC command for a partitioned table space.

**Figure 98: -SPAC command—partitioned table space**

```
-SPAC 001500 DDNAME C001 BMCL TS C7PCF1.TS34P4
-SPAC 001550 DDNAME D001 BMCL TS C7PCF1.TS34P4
-SPAC 001600 DDNAME E001 BMCL TS C7PCF1.TS34P4
-SPAC 001650 DDNAME F001 BMCL TS C7PCF1.TS34P4
```

Figure 99 on page 338 shows an example of the -SPAC command for a nonpartitioned table space and index.

**Figure 99: -SPAC command—nonpartitioned table space and index**

```
-SPAC 001300 DDNAME SYCL0001 BMCC TS C7PCB1.TS13S
-SPAC 001800 DDNAME SYCL0001 BMCC IX C7PCB1C.IX33A3
```

**-SPBX (External SQL stored procedures)**

The -SPBX command is used to create stored procedures that are written in SQL Procedures language (SPL).

The command includes the CREATE PROCEDURE statement and compile, link, and bind options.

- The SQLID option specifies the current SQLID.
- The BUILDSPROC option identifies the stored procedure that Execution must invoke to build the SPL procedure. The default is SYSPROC.DSNTPSMP.
- The PCOPTS option specifies the information that is necessary to precompile the SQL statements.
- The CMOPTS option specifies the information that is necessary to compile the SQL statements.
The PLKDOPTS option specifies the information that is necessary to prelink the application.

The LKEDOPTS option specifies the information that is necessary to link the application.

The BINDOPTS option specifies the information that is necessary to bind the application.

---

**Note**

If the stored procedure builder routine fails after a stored procedure is added to the DB2 catalog, an execution restart will not be successful. Before proceeding, you must manually drop the stored procedure and delete the source and options from the DB2 catalog.

---

Figure 100 on page 339 shows an example of the -SPBX command.

**Figure 100: -SPBX command**

```sql
-SPBX 000350 CREATE PROCEDURE TESTSPBX.SPSQ000A
  ( INOUT IOSF1CHAR3U          CHAR (3) FOR SBCS DATA CCSID EBCDIC
  , INOUT IOSF1CHAR10U         CHAR (11) FOR SBCS DATA CCSID EBCDIC
  , INOUT IOSF1INTEGERU        INTEGER
  , INOUT IOSF1SMALLINTU       SMALLINT
  , INOUT IOSF1FLOAT1U         FLOAT
  , INOUT IOSF1FLOAT22U        FLOAT
  , INOUT IOSF1REALU           FLOAT
  , INOUT IOSF1DOUBLEU         FLOAT
  , INOUT IOSF1DEC102U         DECIMAL (10,2)
  , INOUT IOSF1VARCHAR10       VARCHAR (10) FOR SBCS DATA CCSID EBCDIC
  , INOUT IOSF1DATEU           DATE
  , INOUT IOSF1TIMEU           TIME
  , INOUT IOSF1TIMESTAMPU      TIMESTAMP
  , INOUT IOSF1DELIMU          DECIMAL (10,2)
  )
DYNAMIC RESULT SET 0
LANGUAGE SQL
PARAMETER STYLE GENERAL WITH NULLS
FENCED
NOT DETERMINISTIC
CALLED ON NULL INPUT
MODIFIES SQL DATA
NO DBINFO
NO COLLID
WLM ENVIRONMENT DEEG1
ASUTIME NO LIMIT
STAY RESIDENT NO
PROGRAM TYPE MAIN
SECURITY DB2
COMMIT ON RETURN NO
PARAMETER CCSID EBCDIC
EXTERNAL NAME 'SPBXSP0A'
-- BEGINNING OF SPL PROGRAM
  P1: BEGIN NOT ATOMIC
  DECLARE MANAGER_ID          CHAR(6);
  DECLARE BONUSPRM            DECIMAL(15,2)   DEFAULT 0;
  -- GIVE THE MANAGER A RAISE
  UPDATE EMP
    SET BONUS = BONUSPRM
    WHERE EMPNO = MANAGER_ID;
```

Chapter 5 Worklist commands 339
-SQL (SQL statement)

The -SQL command consists of an SQL statement and executes using dynamic SQL.

All SQL statements except the SELECT statement work with this command. The -SQL command creates and alters external stored procedures. Figure 101 on page 340 shows an example of the -SQL command.

Figure 101: -SQL command

```
-SQL 000170
CREATE TABLESPACE PROPERTY
   IN RESTATE
   USING
      STOGROUP DB2030
      PRIQTY 3
      SECQTY 3
      ERASE NO
      FREEPAGE 5
      PCTFREE 25
      BUFFERPOOL BP0
      LOCKSIZE PAGE
      CLOSE YES
```

**Note**

Do not use the -SQL command for setting an SQL ID or current path. Restart processing might not set the SQLID correctly if an SQL command is used.

When you create a data-only migrate worklist, the Analysis component generates and groups all of the -SQL DELETE commands before the -LOAD commands. These commands appear in phase 2 of the worklist, which is indicated by the -MIGR PHASE-2 command.

Figure 102 on page 340 shows an example of an -SQL command that alters and regenerates a native SQL stored procedure.

Figure 102: -SQL command—ALTER PROCEDURE

```
-SQL 001450
ALTER PROCEDURE MG017D.SPSQN018
   REGENERATE VERSION V1
```
Figure 103 on page 341 shows an example of an -SQL command that adds a comment to a native SQL stored procedure.

**Figure 103: -SQL command—COMMENT**

```
-SQL 000600 COMMENT ON PROCEDURE MG017R.SPSQN047
     VERSION V1
     IS 'THIS IS VERSION V1'
```

**-SQLP (Native SQL stored procedures)**

The -SQLP command creates and alters native SQL stored procedures.

Figure 104 on page 341 shows an example of an -SQLP command that creates a version of a procedure.

**Figure 104: -SQLP command—CREATE PROCEDURE**

```
-SQLP 000700
     CREATE PROCEDURE MG017D.SPSQN017
     (INOUT MEDSAL FLOAT
     )
     VERSION V1
     DYNAMIC RESULT SETS 8
     LANGUAGE SQL
     PARAMETER CCSID ASCII
     NOT DETERMINISTIC
     CALLED ON NULL INPUT
     MODIFIES SQL DATA
     WLM ENVIRONMENT FOR DEBUG MODE AAA
     ASUTIME LIMIT 50
     DISALLOW DEBUG MODE
     COMMIT ON RETURN NO
     QUALIFIER ABC
     PACKAGE OWNER RDAMCG
     SQL PATH "RDAMCG"","RDAMCG2"
     OPTHINT 'XXX'
     DEFER PREPARE
     CURRENT DATA NO
     DYNAMICRULES RUN
     APPLICATION ENCODING SCHEME ASCII
     WITH EXPLAIN
     WITH IMMEDIATE WRITE
     WITH KEEP DYNAMIC
     ISOLATION LEVEL CS
     RELEASE AT DEALLOCATE
     REOPT ONCE
     VALIDATE RUN
     APPLCOMPAT V12R1M500
     ARCHIVE SENSITIVE YES
     BUSINESS_TIME SENSITIVE YES
     SYSTEM_TIME SENSITIVE YES
     ROUNDED DEC_ROUND_CEILING
     DECIMAL(15)
```
Figure 105 on page 342 shows an example of an -SQLP command that adds a version to an existing native SQL stored procedure.

**Figure 105: -SQLP command—ADD VERSION**

```
-SQLP 000550
   ALTER PROCEDURE MG017D.SPSQN017
       ADD VERSION V1
       (IN    P_PACKAGE CHAR (6) FOR SBCS DATA CCSID EBCDIC
        , INOUT P_TS CHAR (26) FOR SBCS DATA CCSID EBCDIC
       )
       DYNAMIC RESULT SET 1
       LANGUAGE SQL
```

Figure 106 on page 342 shows an example of an -SQLP command that specifies the active version of the procedure.

**Figure 106: -SQLP command—ACTIVATE VERSION**

```
-SQLP 000800
   ALTER PROCEDURE MG017R.SPSQN047
       ACTIVATE VERSION V3
```

**-SSID (Subsystem ID)**

The -SSID command identifies the DB2 subsystem where the worklist is processed.

The initial SSID is the DB2 system that is generating the worklist. The SSID should change only for generated migrate worklists. Figure 107 on page 342 shows an example of the -SSID command.

**Figure 107: -SSID command**

```
-SSID 000100 DBAH
```

The -SSID command can also specify a DB2 data sharing group attachment name for the DB2 subsystem ID.

In the worklist, you can enter an asterisk (*) in place of the SSID name. The asterisk indicates to use the SSID that is specified with the SSID keyword in the AEXIN input stream. You can manually specify the SSID as the second value of the PARM parameter of an EXEC statement in a cataloged procedure (which begins with a PROC statement) in the JCL. By doing so, you can run the JCL several times on different subsystems, without having to use a different data set for each subsystem. When you manually specify the SSID, its value overrides the value of the SSID keyword in the AEXIN input stream.

The CM/PILOT component of CHANGE MANAGER also uses the -SSID command in a CM/PILOT worklist.
**-STOP (Stop)**

The **-STOP** command causes execution to stop, commit all work that is performed, and to write a record to the sync table.

This point then becomes the restart point for the next execution of the worklist. The text of the command is a user comment that is included in the sync record. Figure 108 on page 343 shows an example of the **-STOP** command.

**Figure 108: -STOP command**

```
-AUTH 000160 JEO
-SQL 000170
   CREATE TABLESPACE PROPERTY
      IN RESTATE
     USING
       STOGROUP DB2030
       PRIQTY 3
       SEQQTY 3
       ERASE NO
       FREEPAGE 5
       PCTFREE 25
       BUFFERPOOL BP0
       LOCKSIZE PAGE
     CLOSE YES
-STOP 000171 STOP AFTER CREATING TABLESPACE
```

**-STOP** commands are useful for dividing worklist processing into manageable units or for setting points at which you can review progress. This command is generated before creating table spaces and indexes that are specified with the VCAT parameter. This action allows you to execute any allocation changes to the VSAM data sets.

When a worklist is executed in parallel in the Database Administration or BMC Object Administration for DB2 solution, the **-STOP** command is enclosed between the **-BEGU** and **-ENDU** worklist commands.

**-SYNC (Sync point)**

The **-SYNC** command creates a sync table entry and executes a DB2 COMMIT WORK statement.

The text of the command includes a comment that is recorded in the sync table. The Analysis component creates these commands at required intervals in a worklist. Analysis also uses the **SYNCPOINT** keyword in the ALUIN input stream to create additional **-SYNC** commands in a worklist based on the number of **-SQL** commands since the last **-SYNC** command. Figure 109 on page 343 shows an example of the **-SYNC** command.

**Figure 109: -SYNC command**

```
-RNST 000630 RUNSTATS TABLESPACE RESTATE.PROPERTY INDEX (ALL)
```
-SYNC 000640 CHECKPOINT AFTER RUNSTATS
********************************************************
-AUTH 000690 JEO
-REBD 000700 REBIND PLAN ALUAPL1
   VALIDATE(BIND) ISOLATION(CS)
   ACQUIRE(USE) RELEASE(COMMIT) EXPLAIN(NO)
-AUTH 000710
-SYNC 000720 END OF REBIND PLANS SECTION
********************************************************
-COPY 000730
COPY TABLESPACE RESTATE.PROPERTY
DEVT TAPE COPYDDN SYSC1001
-SYNC 000740 CHECKPOINT AFTER IMAGE COPY
********************************************************

**Note**

The product might truncate the name of an object in the -SYNC command if the name contains up to 128 characters. To determine the full name of the object, review the worklist command that precedes the -SYNC command in the worklist.

A -STOP command records a RESTART point and halts processing, while a -SYNC command records a RESTART point and continues processing. Because they mark RESTART points, -SYNC commands should not be removed.

You can insert additional parameters as allowed by DB2 in this command. Execution passes these parameters unchanged to the DB2 utility.

The CM/PILOT component of CHANGE MANAGER also uses the -SYNC command in a CM/PILOT worklist. The sync table entries are created during CM/PILOT worklist processing. You can view the sync table entries for task IDs.

A -SYNC command includes one of the following values in a CM/PILOT worklist:

- END OF ANALYSIS SECTION
- END OF BASELINE REPORT SECTION
- END OF BASELINE SECTION
- END OF BATCH JCL SECTION
- END OF COMPARE SECTION
- END OF DELETE WORKID SECTION
- END OF IMPORT SECTION
- END OF REPLICATE WORKID SECTION
-TIME (File creation time)

The -TIME command defines the time and date that the worklist was created.

-TIME must be the first command in the worklist file. The parameter for -TIME is a 26-byte string, in DB2 TIMESTAMP format. Figure 110 on page 345 shows an example of the -TIME command.

Figure 110: -TIME command

```
-TIME 000050 '2010-05-06 11:02:41.0362'
```

The CM/PILOT component of CHANGE MANAGER also uses the -TIME command in a CM/PILOT worklist.

-UNLI (IBM UNLOAD utility)

The -UNLI command invokes the IBM UNLOAD utility to unload data that the IBM LOAD utility can load.

The utility runs as a subtask. You can use the IBM UNLOAD utility to perform the following functions:

- Dynamically allocate unload data sets
- Use multitasking
- Unload data in LOB columns and XML columns
- Migrate tables with ROWID columns

For more information, see the IBM documentation.

Use of -UNLI to dynamically allocate unload data sets

When you specify to use the IBM UNLOAD and LOAD utilities, the utilities can dynamically allocate the unload data sets.

When Analysis generates the -UNLI command, it specifies a TEMPLATE descriptor of SYRC for the unload (SYSREC) data sets. The DSN parameter in the descriptor specifies the fully qualified data set name for the object that is unloaded. Figure 111 on page 345 shows an example of a -UNLI command that uses dynamic allocation.

Figure 111: -UNLI command—dynamic allocation of data sets

```
-UNLI 000350
 TEMPLATE SYRC
```
DSN ’RDACRJ.MIG2.SR000001’
UNIT SYSDA

UNLOAD FROM TABLE B8P254A.T_EMPLOYEE_TCP
HEADER NONE
UNLDDN SYRC
EBCDIC CCSID(37,0,0)
NOPAD MAXERR 1

Analysis generates the MAXERR parameter with a default value of 1 for the UNLOAD statement. The utility stops when it encounters the first error. The utility does not use a discard data set.

Note
If the IBM UNLOAD utility fails while unloading the data, you must manually delete the appropriate unload (SYSREC) data sets before restarting the worklist. IBM UNLOAD does not automatically delete the dynamically allocated data sets.

Use of -UNLI to multitask

When you specify to use the IBM UNLOAD and LOAD utilities, the utilities can multitask the unloading and loading of data.

For multitasking to occur, a partitioned table space must be specified and the unload (SYSREC) data set must be dynamically allocated. Figure 112 on page 346 shows an example of a -UNLI command that uses multitasking.

Figure 112: -UNLI command—multitasking a partitioned table space

-UNLI 000350
TEMPLATE SYRC
DSN ’RDACRJ.MIG.SR000001.P&PART.’
UNIT SYSDA

UNLOAD FROM TABLE B8P1024A.TB1024
HEADER NONE
UNLDDN SYRC
EBCDIC CCSID(37,0,0)
NOPAD MAXERR 1

Analysis uses the P&PART symbolic variable in the DSN parameter to ensure that the data set name for each partition is unique.

Use of -UNLI to unload data in LOB and XML columns

The IBM UNLOAD utility can be used to unload data that is contained in LOB and XML columns.

The utility requires that the file reference partitioned data sets be dynamically allocated and on DASD.
In addition to the SYRC TEMPLATE descriptor, the -UNLI command specifies a TEMPLATE descriptor of SY xxnn for file reference data sets that contain LOB or XML data. The xx represents LB for LOB data or XC for XML data, and nn represents a sequential number for each LOB or XML column in the table. Each TEMPLATE descriptor includes the directory blocks, and primary and secondary space. Analysis obtains these values from the JCL Generation product options file (POF).

**Note**

IBM recommends running the RUNSTATS utility before using TEMPLATE descriptors and dynamically allocating data sets. However, if a table does not contain statistics, IBM UNLOAD might fail. When Analysis generates a worklist, it determines whether statistics are available on the table. If statistics are found, Analysis removes the directory block, and primary and secondary space allocations.

Figure 113 on page 347 shows an example of a -UNLI command that unloads XML data.

**Figure 113: -UNLI command—unloading XML data**

```
-UNLI 000350
  TEMPLATE SYRC
    DSN 'RDACRJ.MXML.SR000001'
    UNIT SYSDA
  TEMPLATE SYXC01
    DSN 'RDACRJ.DBDC.MXML.SR000001.X01'
    UNIT SYSDA
    DIR 250 SPACE(10,2) CYL
  TEMPLATE SYXC02
    DSN 'RDACRJ.DBDC.MXML.SR000001.X02'
    UNIT SYSDA
    DIR 250 SPACE(10,2) CYL
  UNLOAD FROM TABLE MGX02NG.T27N01LONG
    HEADER NONE
    (TABKEY
     .CHAR5
     .CHAR10
     .DEC15
     .DEC15_3
     .VAR_CHAR15
     .XMLCOL1 VARCHAR CLOBF SYXC01
     .XMLCOL2 VARCHAR CLOBF SYXC02
     .TIMESTAMP)
  UNLDDN SYRC
  EBCDIC CCSID(37,0,0)
  NOPAD MAXERR 1
```

Each LOB or XML column in the column list portion of the UNLOAD statement includes the SY.xxnn TEMPLATE DESCRIPTOR. The column list does not include the DB_GENERATED_DOCID_FOR_XML column.
Use of -UNLI to migrate tables with ROWID columns

You can modify or migrate tables that contain ROWID columns with data.

If the ROWID column is defined as GENERATED ALWAYS, ALTER and CHANGE MANAGER omit the ROWID column in the UNLOAD and LOAD statements. If the ROWID column is defined as GENERATED BY DEFAULT, ALTER and CHANGE MANAGER include the ROWID column in the UNLOAD and LOAD statements. The column is unloaded last in the sequence of columns and is also loaded last, even though it might not be the last column in the table. When a table that contains a ROWID column is unloaded, Analysis lists all of the columns that are unloaded in the UNLOAD statement in the worklist.

-UNRC (Unload record)

Analysis generates the -UNRC command when it creates a worklist to establish a full-recovery baseline.

-UNRC records the names of unload data sets that are used for recovery purposes. Figure 114 on page 348 shows an example of the -UNRC command.

Figure 114: -UNRC command

```
-BMCD 000300 SELECT * FROM GEC.ADDRESS ;
   DDNAME BLRP1001
-UNRC 000350 GEC.ADDRESS
   DDNAME BLRP1001 RECORDID 0000
```

The -UNRC command appears immediately after the -BMCD or -UNLI command that performs the unload. The DDNAME parameter identifies the DDName where the data for the specified object is unloaded. Execution identifies the name of the data set and writes it to the CHANGE MANAGER CM_UNLOADDSN table.

Note

When the unload data set is allocated dynamically, the DSNAME parameter identifies the data set name, as shown in Figure 115 on page 348. The DDNAME parameter is not used.

Figure 115: -UNRC command—dynamic allocation of data set

```
-BMCD 000550
   UNLOAD UNLOADDN (SYSREC01)
   ACTIVE (YES,NO)
   DISCARDS 1
   NULLTYPE T1 NULLCHAR ?
   SELECT '0001', COLUMN_1B, COLUMN_2B
   INTO BMCCONS_RECORDID CHAR(4),
         COLUMN_1B, COLUMN_2B
   FROM ACMU01.T_U07NSB
```
If statistics indicate that a table is empty and the NOUNLOADEMPTY keyword is specified in the ALUIN input stream, the -UNRC command specifies the EMPTY parameter, as shown in Figure 116 on page 349. The command records that the table is included in the baseline but was not unloaded because it is empty.

**Figure 116: -UNRC command for an empty table**

```
-UNRC 000300 GEC.ADDRESS
       EMPTY
```

If the underlying VSAM data set for the table space does not exist, the -UNRC command specifies the DEFINENO parameter, as shown in Figure 117 on page 349.

**Figure 117: -UNRC command for a nonexistent data set**

```
-UNRC 000300 GEC.ADDRESS
       DEFINENO
```

If the unload data set contains XML or LOB data, the -UNRC command specifies the FILEREF parameter, as shown in Figure 118 on page 349.

**Figure 118: -UNRC command for a file reference**

```
-UNRC 000450 MGX01.TBXML
       RECORDID 0000
       FILEREF
       DSNAME TPNS04.DBDC.FULLREC3.BL000001
```

The -UNRC command is used with the -JCLP command (although never in the same worklist). While the -UNRC command records the names of recovery data sets when a baseline is established, the -JCLP command generates proper JCL DD statements to reload those data sets during database recovery of a full-recovery baseline.
Note
If you are restarting a worklist that contains -BMCD or -UNLI commands and -UNRC commands, ensure that the commands use identical data set names. This confirmation is important because these names are being recorded in a full-recovery baseline. If you must restart -BMCD or -UNLI with a different data set name, modify -UNRC to record the correct data set name.

-WKID (Work ID)

The -WKID command identifies the work ID that generated the worklist.

This command must match the work ID in the execution control record. Figure 119 on page 350 shows an example of the -WKID command.

Figure 119: -WKID command
WKID 000300 GEC.BRLOAD2

Use of -WKID in CM PILOT

The CM/PILOT component of CHANGE MANAGER also uses the -WKID command in a CM/PILOT worklist.

In CM/PILOT, the -WKID command creates a new work ID.

WARNING
If you reuse the same work ID, the change definition (CD) table entries for the existing work ID are deleted before the new work ID is created.

If you constantly run the same job, you can simply edit the CM/PILOT worklist, change the work ID name that is specified with the -WKID command, and resubmit the job. To specify a name for the work ID, you can use a name template. (For more information, see the ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2.)

You can also reuse the same work ID by starting the CM/PILOT worklist over, provided that the following conditions exist:

- The task ID has a status of Exec Strt or Exec Comp
- The work ID has a status of Created, Analyzed, or Exec Comp

If you want to prevent the automatic reuse of an existing work ID by the -WKID command, use the NOWKIDREPLACE keyword in the AEXIN input stream of the JCL that processes the CM/PILOT worklist, as shown in Figure 120 on page 351.
You must manually edit and add the keyword to the JCL. If the work ID exists, processing stops and an error message is generated in the diagnostic output. If the work ID does not exist, the work ID is created and processing continues.

**Figure 120: NOWKIDREPLACE keyword**

```plaintext
//AEXIN DD *
AUC
SSID DBAH TASKID DEM.TASK920 STARTOVER
NOWKIDREPLACE
DASDDOPT DS712BDC
```

- **ZPRM (ZPARAM value)**

The –ZPRM command lists an IBM DB2 ZPAM and its value.

You can list the following DSNZPARMS in the command:

- `DDL_MATERIALIZATION`
- `PREVENT_NEW_IXCTRL_PART`

**Example**

```plaintext
-WKID 000150 RDACRJ.RC10503A
-ZPRM 000200 PREVENT_NEW_IXCTRL_PART NO
-SYNC 000250 END OF CHECK ZPARM SECTION
```

**CM/PILOT worklist commands**

This section describes each of the commands that the CM/PILOT component uses in a worklist.

You can edit the CM/PILOT worklist if you need to add or modify commands, keywords, or parameters.

The CHANGE MANAGER components, keywords, and parameters that are associated with a command are described in detail in the “File format” on page 259.

CM/PILOT also uses the following CHANGE MANAGER worklist commands:

- `-BASE`
- `-JCLP`
- `-SSID`
-SYNC
-TIME
-WKID

For a description of these commands, see “ALTER and CHANGE MANAGER worklist commands” on page 261.

-ANLY (Analysis)

The -ANLY command invokes the Analysis component of CHANGE MANAGER.

Analysis evaluates change and migrate requests for a number of issues. If successful, Analysis then generates a CHANGE MANAGER worklist. Figure 121 on page 352 shows an example of the -ANLY command.

**Figure 121: -ANLY command**

```
-ANLY 000600
CMPILOT REUSEWORKID
TRIAL
SSID DBAH
WORKID RDAMAF2.CHGDBDML2
INCLUDE (DATA AMS SQL REBIND)
BMCCHECK BMCREORG
PIC NOSTATS
NOCOPY BMCLLOAD ALTUNLOAD
```

The following CM/PILOT keywords can appear as part of the -ANLY command:

**CMPILOT**

This keyword specifies that DML processing is needed. Do not delete this keyword if you are processing a task ID with DML.

**REUSEWORKID**

This keyword specifies the use of the work ID that is listed in the previous -WKID command. Do not delete this keyword if you are processing a task ID with DML.

**TRIAL**

This keyword specifies that only a DML trial report is generated. The CM/PILOT worklist contains the necessary commands to produce the report. A CHANGE MANAGER worklist is not created.
Upon successful completion, CM/PILOT populates the product CD tables for the work ID. Before running Analysis, you can review or modify any of the changes in Specification.

-BJCL (Batch Execution JCL Generation)

The -BJCL command invokes the Batch Execution JCL Generation function of CHANGE MANAGER.

The Batch Execution JCL Generation function creates the CHANGE MANAGER execution JCL that is necessary to process the CHANGE MANAGER worklist. Figure 122 on page 353 shows an example of the -BJCL command.

**Figure 122: -BJCL command**

```plaintext
-BJCL 001200

PRODUCT ACM
WORKID DEM.TEST1020
SSID DEBF
VERS7FMT Y
DATASETSIZING NONE
DASDDOPT DS613EBF
CATDOPT DC712EBF
NOCATRECOVER
WORKLISTDDN WORKL001
```

-COMP (Compare)

The -COMP command invokes the Compare component of CHANGE MANAGER.

This component is used to compare two input sources and creates a Change Definition Language (CDL) file. The CDL file contains commands that, if applied to one input source, would create the data structures of the other input source. Figure 123 on page 353 shows an example of the -COMP command.

**Figure 123: -COMP Command**

```plaintext
-COMP 000700

SSID DBAH
CMPTYPE1 BASELINE
CMPIN1T DEM.DDLTEMPLATE
CMPTYPE2 BASELINE
CMPIN2 DEM.BASE1234
INCLUDE (ALTER CREATE DROP )
```

The following CM/PILOT keywords can appear as part of the -COMP command:

- **CMPIN1T**

  You can use a predefined script or a user-defined script to create a baseline and use a baseline name template to name the baseline. In the script, the
CMPIN1T keyword is used instead of the CMPIN1 keyword for the primary input source of a comparison. The parameter for the CMPIN1T keyword is the baseline profile.

When the CM/PILOT worklist is processed, the most recent baseline created with the baseline profile is used for the primary input source for the comparison step.

**CMPIN2T**

You can use a predefined script or a user-defined script to create a baseline and use a baseline name template to name the baseline. In the script, the CMPIN2T keyword is used instead of the CMPIN2 keyword for the secondary input source of a comparison. The parameter for the CMPIN2T keyword is the baseline profile.

**-DML (DML section)**

The -DML command identifies the section of the CM/PILOT worklist that contains DML statements.

*Figure 124 on page 354* shows an example of the -DML command.

**Note**

No other CM/PILOT worklist commands can exist between the -ANLY command and the -DML command.

All of the DML statements for a task ID must be contained in one DML section. Multiple -DML commands cause an error.

*Figure 124: -DML command*

```
-DML 000700
UPDATE DATABASES SET
  NAME           =    'KIF'
WHERE
  NAME           LIKE 'MAF%'
;
UPDATE TABLESPACES SET
  NAME           =    'ACTVEN'
WHERE
  NAME           =    'ACT'
```

**-IMP (Import)**

The -IMP command invokes the Import component of CHANGE MANAGER.
Import receives CDL files, DDL files, and CHANGE MANAGER worklists from another subsystem and imports them to the local subsystem so that their changes can be applied. Figure 125 on page 355 shows an example of the -IMP command.

**Figure 125: -IMP command**

```
-IMP 001000
SSID DBAH
REPLACEWORKID RDAMAF2.RECSR
SOURCETYPE CDL
IMPORTINDN CDL001
```

**Note**

When the CM/PILOT component builds the CM/PILOT worklist, the component changes the WORKID ALUIN keyword to the REPLACEWORKID ALUIN keyword.

-**OPTS (Options)**

The -OPTS command identifies the keywords that the REPLICATE WORKID, DELETE WORKID, and DELETE SYNCTABLE DML statements use.

The command does not perform any action during worklist processing. The -OPTS command must precede the -REPL command in a CM/PILOT worklist. Figure 126 on page 355 shows an example of the -OPTS command for the DELETE WORKID DML statement.

**Figure 126: -OPTS command**

```
-OPTS 000500
SSID DBDC
NOREPLACE
TRIAL
-REPL 000600
DELETE WORKID
SET WKOWNER='RDACRJ'
WHERE WKOWNER='CRJ';
```

The following CM/PILOT keywords can appear as part of the -OPTS command:

**SSID**

This keyword identifies the DB2 subsystem ID or the DB2 data sharing group attachment name.

**NOREPLACE**

For the REPLICATE WORKID DML statement, this keyword specifies that existing work IDs will not be replaced.

You cannot specify NOREPLACE and REPLACE in the same -OPTS command.
REPLACE

For the REPLICATE WORKID DML statement, this keyword specifies that existing work IDs will be replaced.

You cannot specify NOREPLACE and REPLACE in the same -OPTS command.

TRIAL

This keyword specifies that only a DML trial report is generated. The CM/PILOT worklist contains the necessary commands to produce the report.

You must manually add this keyword to the CM/PILOT worklist.

DEBUG

This keyword causes CM/PILOT to print complete diagnostic information in the diagnostic output file.

This keyword might be necessary if you suspect that CM/PILOT is operating incorrectly and you need to send supporting documentation to BMC for research.

You must manually add this keyword to the CM/PILOT worklist.

-REPL (REPL section)

The -REPL command identifies the section of the CM/PILOT worklist that contains the REPLICATE WORKID, DELETE WORKID, or DELETE SYNCTABLE DML statement.

Figure 127 on page 356 shows an example of the -REPL command for the REPLICATE WORKID DML statement.

Figure 127: -REPL command

```
-REPL 000600
  REPLICATE WORKID
    SET WKOWNER='RIHJCB'
    WHERE WKOWNER='JCB'
    AND WKNAME='BWSDB001';
  CHANGE DATABASES
    SET DBNAME='AAMX1001'
    WHERE DBNAME='AALURU34';
```
-TASK (Task ID)

The -TASK command identifies the task ID that is associated with the CM/PILOT worklist.

Figure 128 on page 357 shows an example of the -TASK command.

![Figure 128: -TASK command](image)

-TASK 000300 RDAMAF2.CHGDBDML

Other worklist commands

This section describes worklist commands that the Execution component can use, but that are not generated by the product.

-BMCU (Execute a BMC utility)

The -BMCU command invokes a BMC utility to run as a subtask.

Figure 129: -BMCU command

![Figure 129: -BMCU command](image)

-BMCU 000004 ASUSMAIN
BMCSTATS INDEX QZU.QZUX01_D30S05T01
EVENTS N

**Note**

The utility command starts on the second line of the statement. The first line is reserved for parameters passed to the utility, such as NEW, TERM, MAINT, NEW/RESET, and TERM/RESET.

The text portion of the statement identifies the invoked program name and the parameters passed to the program.

DASD MANAGER PLUS generates this command.

-DEBUG (Debug)

The -DEBUG command turns on or off the DEBUG and FLOW options.

The following keywords are valid:

- DEBUG
- DEBUGOFF
The Administrative products do not generate this command. If necessary, contact BMC Customer Support, who will direct you to add this command to your worklist to resolve an issue.

-DSN1 (IBM DSN1COPY utility)

The -DSN1 command invokes the IBM DSN1COPY utility.

If you need to code this command yourself (see Figure 130 on page 358), use the following rules:

- The **Command** line (the one with the -DSN1 command on it) must contain only the sequence number and the command.

- The second statement should contain the DD keyword DSN1DDIN followed by a ddname.

- The third statement should contain the input DSNAME for DSN1DDIN.

- The fourth statement should contain the output DD keyword DSN1DDOU, followed by a ddname for the output data set that is defined in the JCL. If the output DSN begins with a period, the DSN is storage group defined, and the Execution component looks up the high-level qualifier. (Parm='parameterString').

- Do not define a SYSUT1 DD in the JCL.

**Figure 130: -DSN1 command**

```
-DSN1 000004
DSN1DDIN DS1I0001
DEAEACAT.DSNDBD.WZ2130.WZS12130.I0001.A237
DSN1DDOU DS1O0001
PARM='PAGESIZE(4K)'
```

For more information, see the documentation for the IBM utilities.

**Note**
CATALOG MANAGER and DASD MANAGER PLUS generate this command.
-MERG (IBM MERGECOPY utility)

The -MERG command invokes the IBM MERGECOPY utility.

This command and its parameters are passed to DB2.

**Figure 131: -MERG command**

```
-MERG 050000
    MERGECOPY TABLESPACE BMCASU.BMCUT01
    DEV SYSDA
    COPYDDN SYSCO004
```

For more information, see the documentation for the IBM utilities.

---

**Note**

CATALOG MANAGER and DASD MANAGER PLUS generate this command.

---

-MODI (IBM MODIFY utility)

The -MODI command invokes the IBM MODIFY RECOVERY or MODIFY STATISTICS utility.

**Figure 132 on page 359** shows an example of the -MODI command for the MODIFY RECOVERY utility.

**Figure 132: -MODI command—MODIFY RECOVERY**

```
-MODI 000001
    MODIFY RECOVERY
    TABLESPACE ACTQX18.ACTS0118
    DELETE AGE(*)
```

**Figure 133 on page 359** shows an example of the -MODI command for the MODIFY STATISTICS utility.

**Figure 133: -MODI command—MODIFY STATISTICS**

```
-MODI 000002
    MODIFY STATISTICS
    TABLESPACE ACTQX18.ACTS0118
    DELETE ALL AGE(*)
```

For more information, see the documentation for the IBM utilities.

---

**Note**

CATALOG MANAGER and DASD MANAGER PLUS generate this command.
**-NOOP (No operation)**

The -NOOP worklist command bypasses the statement that follows the command and proceeds to the next worklist command.

*Figure 134 on page 360* shows an example of the -NOOP command.

**Figure 134: -NOOP command**

```
-NOOP 000010
SELECT
  LENGTH( DBNAME ) AS DBLENG
  DBNAME
  , LENGTH( NAME ) AS TSLENG
  , NAME
  , PARTITIONS
  , CURRENT SERVER AS CURRSERV
  , TS.
FROM SYSSIBM.SYSTABLESPACE TS
WHERE DBNAME = 'ACTQX11' AND NAME = 'ACTS0311'
WITH UR
```

*Note*

CHANGE MANAGER generates this command.

**-QUI (IBM QUIESCE utility)**

The -QUI command invokes the IBM DB2 QUIESCE utility.

Following is an example of the -QUI command.

**Figure 135: -QUI command**

```
-QUI 001020
QUIESCE TABLESPACE ASUDBU04.U041
  TABLESPACE ASUDBU04.U042
  TABLESPACE ASUDBU04.U051
  TABLESPACE ASUDBX1.WS11
  TABLESPACE ASUDBX1.WS32
```

For more information, see the documentation for the IBM utilities.

*Note*

CATALOG MANAGER and DASD MANAGER PLUS generate this command.
-REPO (IBM REPORT utility)

The -REPO command invokes the IBM DB2 REPORT utility on table spaces or table space sets as required.

The product passes this command and its parameters to DB2. Figure 136 on page 361 shows an example of the -REPO command.

Figure 136: -REPO command

```
-REPO 000001
   REPORT RECOVERY TABLESPACE
   ACTQX16.ACTS0116
   CURRENT
   SUMMARY
   LOCALSITE
   RECOVERYSITE
```

For more information, see the documentation for the IBM utilities.

**Note**

CHANGE MANAGER and DASD MANAGER PLUS generate this command.

-REPX (IBM REPAIR utility)

The -REPX command invokes the IBM REPAIR utility on table spaces as required.

The product passes this command and its parameters to DB2.

For more information, see the documentation for the IBM utilities.

**Note**

The first line of the -REPX command must be blank.

DASD MANAGER PLUS generates this command.

-SQLM (Group multiple ALTER statements)

The -SQLM command groups multiple ALTER statements (for table space partitions and their indexes) within a single -SQL command to reduce the total number of steps in the worklist and increase worklist performance.

Figure 137: -SQLM command

```
-SQLM 000002 ALTER TABLESPACE ARMDBJTR.TS40P5 PART 001
   PRIOTY 48 SECOTY 720;
   ALTER TABLESPACE ARMDBJTR.TS40P5 PART 002
```
DASD MANAGER PLUS generates this command.

**-STOS (IBM STOSPACE utility)**

The `-STOS` command invokes the IBM STOSPACE utility.

**Figure 138: -STOS command**

```
-STOS 010000
STOSPACE STOGROUP DEV070
```

For more information, see the documentation for the IBM utilities.

Note

CHANGE MANAGER and DASD MANAGER PLUS generate this command.

**Authorizations used for authorization switching**

The Execution component performs an authorization switching function to ensure that DB2 structures are created with the proper owner and creator.

“Authorization switching using non-GLID processing” on page 363 and “Authorization switching using GLID processing” on page 367 show the authorizations that are used when switching authorizations for processing DB2 objects, utilities, and commands.

**Description of columns in authorization switching tables**

The descriptions of the table columns for “Authorization switching using non-GLID processing” on page 363 and “Authorization switching using GLID processing” on page 367 are provided in Table 22 on page 363.
Table 22: Columns in authorization switching tables

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object, utility, or command</td>
<td>The DB2 object, utility, or command that might require an authorization switch</td>
</tr>
<tr>
<td>SQL statement</td>
<td>The SQL statement for the object</td>
</tr>
<tr>
<td>Installation option setting</td>
<td>The setting for the authorization switching option as specified in the installation options module (AUTHSW=Y, AUTHSW=N, AUTHSW=B, AUTHSW=G, or AUTHSW=X)</td>
</tr>
</tbody>
</table>

Description of authorizations used in the switching tables

The descriptions for the authorizations that are shown in “Authorization switching using non-GLID processing” on page 363 and “Authorization switching using GLID processing” on page 367 are provided in Table 23 on page 363.

Table 23: Authorizations used in switching tables

<table>
<thead>
<tr>
<th>Authorization</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bound by</td>
<td>Authorization ID of the user who bound the plan or package</td>
</tr>
<tr>
<td>Created by</td>
<td>Authorization ID of the user who created the object</td>
</tr>
<tr>
<td>GLID</td>
<td>Global authorization ID specified with the -GLID worklist command</td>
</tr>
<tr>
<td>Grantor</td>
<td>Authorization ID of the user who granted privileges to the object</td>
</tr>
<tr>
<td>Job auth ID</td>
<td>Authorization ID of the user running the job</td>
</tr>
<tr>
<td>Owner</td>
<td>Authorization ID of the owner of the object, plan, or package</td>
</tr>
</tbody>
</table>

Authorization switching using non-GLID processing

Authorization switching can be used when a GLID is not specified.

Table 24 on page 364 shows the authorizations that are used when not specifying a GLID.
### Table 24: Authorization switching IDs for non-GLID processing

<table>
<thead>
<tr>
<th>Object, utility, or command</th>
<th>SQL statement</th>
<th>Installation option setting</th>
<th>AUTHSW=B</th>
<th>AUTHSW=Y</th>
<th>AUTHSW=N</th>
<th>AUTHSW=X</th>
<th>No commands</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>AUTHSW=B</td>
<td>AUTHSW=Y</td>
<td>AUTHSW=N</td>
<td>AUTHSW=X</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-AUTH command</td>
<td>-SETS command</td>
<td>-AUTH command</td>
<td>-SETS command</td>
<td>No commands</td>
<td></td>
</tr>
<tr>
<td>Storage group</td>
<td>ALTER</td>
<td>Created by</td>
<td>Owner</td>
<td>Owner</td>
<td>Owner</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Storage group</td>
<td>CREATE</td>
<td>Created by</td>
<td>Owner</td>
<td>Owner</td>
<td>Owner</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Storage group</td>
<td>DROP</td>
<td>Job auth ID</td>
<td>NA</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Storage group</td>
<td>GRANT</td>
<td>Grantor</td>
<td>NA</td>
<td>Grantor</td>
<td>Grantor</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Database</td>
<td>ALTER</td>
<td>Created by</td>
<td>Owner</td>
<td>Owner</td>
<td>Owner</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Database</td>
<td>CREATE</td>
<td>Created by</td>
<td>Owner</td>
<td>Owner</td>
<td>Owner</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Database</td>
<td>DROP</td>
<td>Job auth ID</td>
<td>NA</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Database</td>
<td>GRANT</td>
<td>Grantor</td>
<td>NA</td>
<td>Grantor</td>
<td>Grantor</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Table space</td>
<td>ALTER</td>
<td>Created by</td>
<td>Owner</td>
<td>Owner</td>
<td>Owner</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Table space</td>
<td>CREATE</td>
<td>Created by</td>
<td>Owner</td>
<td>Owner</td>
<td>Owner</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Table space</td>
<td>DROP</td>
<td>Job auth ID</td>
<td>NA</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Table space</td>
<td>GRANT</td>
<td>Grantor</td>
<td>NA</td>
<td>Grantor</td>
<td>Grantor</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Table</td>
<td>ALTER</td>
<td>Created by</td>
<td>Owner</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Table</td>
<td>COMMENT ON</td>
<td>Created by</td>
<td>Owner</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Table</td>
<td>CREATE</td>
<td>Created by</td>
<td>Owner</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Table</td>
<td>DROP</td>
<td>Job auth ID</td>
<td>NA</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Table</td>
<td>GRANT</td>
<td>Grantor</td>
<td>NA</td>
<td>Grantor</td>
<td>Grantor</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Table</td>
<td>LABEL ON</td>
<td>Created by</td>
<td>NA</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Index</td>
<td>ALTER</td>
<td>Created by</td>
<td>Owner</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Index</td>
<td>CREATE</td>
<td>Created by</td>
<td>Owner</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Index</td>
<td>DROP</td>
<td>Job auth ID</td>
<td>NA</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>View</td>
<td>COMMENT ON</td>
<td>Created by</td>
<td>Owner</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>View</td>
<td>CREATE a</td>
<td>Created by</td>
<td>Owner</td>
<td>Job auth ID</td>
<td>Job auth ID</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>View</td>
<td>CREATE b</td>
<td>Created by</td>
<td>Owner</td>
<td>Owner</td>
<td>NA</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

364  *ALTER and CHANGE MANAGER for DB2 Reference Manual*
<table>
<thead>
<tr>
<th>Object, utility, or command</th>
<th>SQL statement</th>
<th>Installation option setting</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>AUTHSW=B</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>-AUTH</strong> command</td>
</tr>
<tr>
<td>View</td>
<td>CREATE</td>
<td>Created by Owner</td>
</tr>
<tr>
<td>View</td>
<td>DROP</td>
<td>Job auth ID NA</td>
</tr>
<tr>
<td>View</td>
<td>GRANT</td>
<td>Grantor NA</td>
</tr>
<tr>
<td>View</td>
<td>LABEL ON</td>
<td>Created by Owner</td>
</tr>
<tr>
<td>Trigger</td>
<td>COMMENT ON</td>
<td>Job auth ID NA</td>
</tr>
<tr>
<td>Trigger</td>
<td>CREATE</td>
<td>NA</td>
</tr>
<tr>
<td>Trigger</td>
<td>DROP</td>
<td>Job auth ID NA</td>
</tr>
<tr>
<td>Synonym</td>
<td>CREATE</td>
<td>Created by Owner</td>
</tr>
<tr>
<td>Synonym</td>
<td>DROP</td>
<td>Created by Owner</td>
</tr>
<tr>
<td>Alias</td>
<td>COMMENT ON</td>
<td>Created by Owner</td>
</tr>
<tr>
<td>Alias</td>
<td>CREATE</td>
<td>Created by Owner</td>
</tr>
<tr>
<td>Alias</td>
<td>DROP</td>
<td>Job auth ID NA</td>
</tr>
<tr>
<td>CHECK DATA (-CHEK command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>BMC CHECK PLUS (-BMCK command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>COPY (-COPY command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>NGT Copy (-BMCC command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>LOAD (-LOAD command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>BMC LOADPLUS (-BMCL command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>Object, utility, or command</td>
<td>SQL statement</td>
<td>Installation option setting</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTHSW=B</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTHSW=Y</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTHSW=N</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTHSW=X</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-AUTH command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-SETS command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-AUTH command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-SETS command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No commands</td>
</tr>
<tr>
<td>REORG (-REOR command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>BMC REORG PLUS (-BMCR command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>RECOVER (-RECV command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>NGT Recover (-BMCV command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>RUNSTATS (-RNST command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>BMCSTATS (-BMCU command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>BMC UNLOAD PLUS (-BMCD command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>BMC BASIC UNLOAD (-BMCB command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>BIND PACKAGE (-BIND command)</td>
<td>NA</td>
<td>Bound by</td>
</tr>
<tr>
<td>BIND PLAN (-BIND command)</td>
<td>NA</td>
<td>Bound by</td>
</tr>
<tr>
<td>REBIND PACKAGE (-REBD command)</td>
<td>NA</td>
<td>Bound by</td>
</tr>
</tbody>
</table>

Authorizations used for authorization switching
<table>
<thead>
<tr>
<th>Object, utility, or command</th>
<th>SQL statement</th>
<th>Installation option setting</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AUTHSW=B</td>
<td>AUTHSW=Y</td>
</tr>
<tr>
<td></td>
<td>-AUTH command</td>
<td>-SETS command</td>
</tr>
<tr>
<td>Storage group</td>
<td>ALTER</td>
<td>Created by</td>
</tr>
<tr>
<td>Storage group</td>
<td>CREATE</td>
<td>Created by</td>
</tr>
<tr>
<td>Storage group</td>
<td>DROP</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>Storage group</td>
<td>GRANT</td>
<td>Grantor</td>
</tr>
<tr>
<td>Database</td>
<td>ALTER</td>
<td>Created by</td>
</tr>
<tr>
<td>Database</td>
<td>CREATE</td>
<td>Created by</td>
</tr>
<tr>
<td>Database</td>
<td>DROP</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>Object, utility, or command</td>
<td>SQL statement</td>
<td>Installation option setting</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTHSW=B</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-AUTH command</td>
</tr>
<tr>
<td>Database</td>
<td>GRANT</td>
<td>Grantor</td>
</tr>
<tr>
<td>Table space</td>
<td>ALTER</td>
<td>Created by</td>
</tr>
<tr>
<td>Table space</td>
<td>CREATE</td>
<td>Created by</td>
</tr>
<tr>
<td>Table space</td>
<td>DROP</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>Table space</td>
<td>GRANT</td>
<td>Grantor</td>
</tr>
<tr>
<td>Table</td>
<td>ALTER</td>
<td>Created by</td>
</tr>
<tr>
<td>Table</td>
<td>COMMENT ON</td>
<td>Created by</td>
</tr>
<tr>
<td>Table</td>
<td>CREATE</td>
<td>Created by</td>
</tr>
<tr>
<td>Table</td>
<td>DROP</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>Table</td>
<td>GRANT</td>
<td>Grantor</td>
</tr>
<tr>
<td>Table</td>
<td>LABEL ON</td>
<td>Created by</td>
</tr>
<tr>
<td>Index</td>
<td>ALTER</td>
<td>Created by</td>
</tr>
<tr>
<td>Index</td>
<td>CREATE</td>
<td>Created by</td>
</tr>
<tr>
<td>Index</td>
<td>DROP</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>View</td>
<td>COMMENT ON</td>
<td>Created by</td>
</tr>
<tr>
<td>View</td>
<td>CREATE a</td>
<td>Created by</td>
</tr>
<tr>
<td>View</td>
<td>CREATE b</td>
<td>Created by</td>
</tr>
<tr>
<td>View</td>
<td>CREATE c</td>
<td>Created by</td>
</tr>
<tr>
<td>View</td>
<td>DROP</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>View</td>
<td>GRANT</td>
<td>Grantor</td>
</tr>
<tr>
<td>View</td>
<td>LABEL ON</td>
<td>Created by</td>
</tr>
<tr>
<td>Trigger</td>
<td>COMMENT ON</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>Trigger</td>
<td>CREATE</td>
<td>NA</td>
</tr>
<tr>
<td>Trigger</td>
<td>DROP</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>Synonym</td>
<td>CREATE</td>
<td>Created by</td>
</tr>
<tr>
<td>Synonym</td>
<td>DROP</td>
<td>Created by</td>
</tr>
<tr>
<td>Object, utility, or command</td>
<td>SQL statement</td>
<td>Installation option setting</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>--------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>AUTHSW=B</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>-AUTH</strong></td>
</tr>
<tr>
<td>Alias</td>
<td>COMMENT ON</td>
<td>Created by</td>
</tr>
<tr>
<td>Alias</td>
<td>CREATE</td>
<td>Created by</td>
</tr>
<tr>
<td>Alias</td>
<td>DROP</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>CHECK DATA (CHECK command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>BMC CHECK PLUS (BMCK command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>COPY (COPY command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>NGT Copy (BMCC command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>LOAD (LOAD command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>BMC LOADPLUS (BMCL command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>REORG (REOR command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
</tbody>
</table>

Chapter 5  Worklist commands  369
<table>
<thead>
<tr>
<th>Object, utility, or command</th>
<th>SQL statement</th>
<th>Installation option setting</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>AUTHSW=B</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTHSW=Y</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTHSW=N</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTHSW=G</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-AUTH command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-SETS command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-AUTH command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-SETS command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-GLID command</td>
</tr>
<tr>
<td>RECOVER (-RECV command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>NGT Recover (-BMCV command)</td>
<td></td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GLID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GLID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NA</td>
</tr>
<tr>
<td>RUNSTATS (-RNST command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>BMCSTATS (-BMCU command)</td>
<td></td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GLID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GLID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NA</td>
</tr>
<tr>
<td>BMC UNLOAD PLUS (-BMCD command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>BMC BASIC UNLOAD (-BMCB command)</td>
<td>NA</td>
<td>Job auth ID</td>
</tr>
<tr>
<td>Command</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIND PACKAGE (-BIND command)</td>
<td>NA</td>
<td>Bound by</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Owner</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GLID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NA</td>
</tr>
<tr>
<td>BIND PLAN (-BIND command)</td>
<td>NA</td>
<td>Bound by</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Owner</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GLID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NA</td>
</tr>
<tr>
<td>REBIND PACKAGE (-REBD command)</td>
<td>NA</td>
<td>Bound by</td>
</tr>
<tr>
<td>REBIND PLAN (-REBD command)</td>
<td>NA</td>
<td>Bound by</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Owner</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GLID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NA</td>
</tr>
<tr>
<td>Object, utility, or command</td>
<td>SQL statement</td>
<td>Installation option setting</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTHSW=B</td>
</tr>
<tr>
<td></td>
<td>-AUTH command</td>
<td>-SETS command</td>
</tr>
</tbody>
</table>

a. The SQL statement applies to creating a view that uses a qualified name for the view.
b. The SQL statement applies to the authorization for unqualified objects specified in the view text of a `CREATE VIEW` statement that uses a qualified name for the view.
c. The SQL statement applies to creating a view that uses an unqualified name for the view.
Change Definition Language

Change Definition Language (CDL) is a BMC proprietary language used to specify changes to DB2 data structures. CDL files transmit data structure changes between subsystems or provide a record of changes to data structures.

CDL files specify changes required to transform a source set of data structures to a target set of data structures. CDL files are used in the change, import, and recovery processes.

File format

Worklist and CDL files consist of a sequence of commands that occupy single or multiple lines.

The file structure is sequential and fixed in length, with 80-character records. Table 19 on page 259 defines the record layout of the worklist file and the record layout of the CDL file.

Table 26: Worklist file format

<table>
<thead>
<tr>
<th>Column number</th>
<th>Layout</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Line identifier</td>
<td>A dash (-) that indicates the beginning of a command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>An asterisk (*) indicates a comment. If there is no indicator, then this</td>
</tr>
<tr>
<td></td>
<td></td>
<td>line is a continuation of the previous Command line.</td>
</tr>
<tr>
<td>2-5</td>
<td>Command identifier</td>
<td>A four-character code that identifies the command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The command identifier appears only on the first line of the command.</td>
</tr>
<tr>
<td>6</td>
<td>Blank</td>
<td>A blank character (required)</td>
</tr>
</tbody>
</table>
## CDL commands

CDL files consist of a sequence of commands.

The commands in a CDL file must appear in the following order:

1. `-TIME`. See “-TIME (File creation time)” on page 345
2. `-ORGN`. See “-ORGN (CDL origin subsystem)” on page 375
3. `-DEST`. See “-DEST (CDL destination subsystem)” on page 375
4. *(optional)* `-GLID`. See “-GLID (Global authorization ID)” on page 311
5. Any number of `-CDL` statements. See “CDL statements” on page 376
-ORGN, -DEST, and -CDL statements are unique to CDL files. The -TIME and -GLID commands are also used in worklists.

-ORGN (CDL origin subsystem)

The -ORGN statement defines the system that originated the CDL.

-ORGN is a required command and must appear after -TIME and before -DEST. Figure 139 on page 375 shows an example of the -ORGN command.

Figure 139: -ORGN command

-ORGN 000100 SSID DBAH LOCATION * RELEASE '101'

The -ORGN command parameters are described as follows:

- Possible values for SSID are the origin subsystem of the CDL statements or an asterisk (*) if the origin subsystem is not known.

- Possible values for LOCATION are the origin location ID of the CDL or * if the origin location is not known.

- RELEASE is the version and release number of the DB2 system from which the CDL was generated. This release string is returned from a DB2 CALL ATTACH and must be enclosed with single quotes.

-DEST (CDL destination subsystem)

The -DEST command defines the intended destination system of the CDL file.

-DEST must be the third command in a CDL file (after -TIME and -ORGN). Figure 140 on page 375 shows an example of the -DEST command.

Figure 140: -DEST command

-DEST 000150 SSID * LOCATION AUSTIN

The -DEST command parameters are described as follows:

- Possible values for SSID are a valid DB2 subsystem identifier, or an asterisk (*) if the subsystem identifier is not known.

- Possible values for LOCATION are a valid DB2 location identifier, or * if the location identifier is not known.

- Both the SSID and the LOCATION clauses are required.
-CDL (CDL statement)

The -CDL command introduces a CDL statement.

Free-format CDL statements follow the statement number on the line that contains the -CDL command and might continue onto as many lines as needed. Figure 141 on page 376 shows an example of the -CDL command.

Figure 141: -CDL command

```
-CDL 000300 DROP TABLE GEC.NAMES
```

CDL reserved words

Several words cannot be used as ordinary identifiers in any context where they could also be interpreted as CDL keywords.

The following words are reserved:

- AS
- CHECK
- DATABASE
- FIELDPROC
- PROCEDURE
- PIECESIZE
- UNIQUE
- DSSIZE

**Note**

You do not need to perform a DROP or REBUILD if a reserved word appears in a column. You can continue processing by delimiting the reserved word with double quotes, for example "AS".

CDL statements

CDL statements are used to create, alter, and drop objects.
The -CDL command in a CDL file contains a CDL statement. Each CDL statement contains a command, parameters, and variables. This section provides a syntax diagram and description of each CDL statement. For information about the conventions that are used in the diagrams, see “Syntax diagrams” on page 13.

Each CDL statement contains variables. Table 27 on page 377 provides a list of the variables and their definitions. The variable definitions use the terms short identifier and long identifier as they are defined in the IBM documentation. A short identifier is 8 characters long. A long identifier is 128 characters long. The ALTER and CHANGE MANAGER products support delimited identifiers for the owners of DB2 objects and DB2 object names that use long identifiers.

Table 27: Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Identifier type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>aliasName</td>
<td>Long</td>
<td>Names an alias</td>
</tr>
<tr>
<td>aliasOwner</td>
<td>Short</td>
<td>Names the owner of an alias</td>
</tr>
<tr>
<td>authorizationID</td>
<td>Short</td>
<td>Specifies an authorization ID that the operating system recognizes as valid</td>
</tr>
<tr>
<td>bufferPoolName</td>
<td>NA</td>
<td>Names a buffer pool</td>
</tr>
<tr>
<td>catalogName</td>
<td>Short</td>
<td>Names a catalog</td>
</tr>
<tr>
<td>checkConstraintName</td>
<td>Short</td>
<td>Names a check constraint</td>
</tr>
<tr>
<td>columnName</td>
<td>Long</td>
<td>Names a column in a table or view</td>
</tr>
<tr>
<td>columnNameList</td>
<td>NA</td>
<td>List of columns for a trigger</td>
</tr>
<tr>
<td>constant</td>
<td>NA</td>
<td>Specifies a value that is passed to a procedure</td>
</tr>
<tr>
<td>constraintName</td>
<td>Short</td>
<td>Names a referential integrity constraint (foreign key definition)</td>
</tr>
<tr>
<td>constraintText</td>
<td>NA</td>
<td>Specifies the check constraint text</td>
</tr>
<tr>
<td>correlationName</td>
<td>NA</td>
<td>Names a correlation</td>
</tr>
<tr>
<td>databaseName</td>
<td>Short</td>
<td>Names a database</td>
</tr>
<tr>
<td>dataTypeName</td>
<td>NA</td>
<td>Names the data type of a table column</td>
</tr>
<tr>
<td>dataTypeSchema</td>
<td>N/A</td>
<td>Names the schema type of a table column</td>
</tr>
<tr>
<td>identifier</td>
<td>NA</td>
<td>Specifies the OLD_TABLE AS or NEW_TABLE AS identifier for a trigger</td>
</tr>
<tr>
<td>indexName</td>
<td>Long</td>
<td>Names an index</td>
</tr>
<tr>
<td>indexOwner</td>
<td>Short</td>
<td>Names the owner of an index</td>
</tr>
<tr>
<td>Variable</td>
<td>Identifier type</td>
<td>Definition</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>locationID</td>
<td>NA</td>
<td>Designates a DB2 subsystem&lt;br&gt;The ID can be 1 to 16 characters in length.</td>
</tr>
<tr>
<td>number</td>
<td>NA</td>
<td>Specifies an integer</td>
</tr>
<tr>
<td>partitionNumber</td>
<td>NA</td>
<td>Identifies a partition in a partitioned index space or table space&lt;br&gt;When the value is 0, it identifies a nonpartitioned index space or table space.</td>
</tr>
<tr>
<td>password</td>
<td>Short</td>
<td>Specifies a password that the operating system recognizes as valid</td>
</tr>
<tr>
<td>procedureName</td>
<td>NA</td>
<td>Names an edit procedure, validation procedure, or field procedure for a table</td>
</tr>
<tr>
<td>schema</td>
<td>NA</td>
<td>Names a collection of objects</td>
</tr>
<tr>
<td>schemaList</td>
<td>NA</td>
<td>Names a list of a collection of objects</td>
</tr>
<tr>
<td>sequenceName</td>
<td>NA</td>
<td>Names a sequence</td>
</tr>
<tr>
<td>statementNumber</td>
<td>NA</td>
<td>Specifies a six-digit statement number in a CDL input file&lt;br&gt;Leading zeros are required, and the statement number must occupy columns 7-12 of the line of a command.</td>
</tr>
<tr>
<td>storageGroupName</td>
<td>Short</td>
<td>Names a storage group</td>
</tr>
<tr>
<td>subselect</td>
<td>NA</td>
<td>Names the SQL subselect statement for a view</td>
</tr>
<tr>
<td>subsystemID</td>
<td>NA</td>
<td>Names a DB2 subsystem&lt;br&gt;The ID can be 1 to 4 characters in length.</td>
</tr>
<tr>
<td>synonymName</td>
<td>Long</td>
<td>Names a synonym</td>
</tr>
<tr>
<td>synonymOwner</td>
<td>Short</td>
<td>Names the owner of a synonym</td>
</tr>
<tr>
<td>tableName</td>
<td>Long</td>
<td>Names a table</td>
</tr>
<tr>
<td>tableOwner</td>
<td>Short</td>
<td>Names the owner of a table</td>
</tr>
<tr>
<td>tableSpaceName</td>
<td>Short</td>
<td>Names a table space</td>
</tr>
<tr>
<td>triggerName</td>
<td>Long</td>
<td>Names a trigger</td>
</tr>
<tr>
<td>triggerOwner</td>
<td>Short</td>
<td>Names the owner of a trigger</td>
</tr>
<tr>
<td>triggerText</td>
<td>NA</td>
<td>Specifies the trigger text</td>
</tr>
<tr>
<td>uniqueConstraintName</td>
<td>NA</td>
<td>Names a unique constraint</td>
</tr>
<tr>
<td>viewName</td>
<td>Long</td>
<td>Names a view</td>
</tr>
<tr>
<td>viewOwner</td>
<td>Short</td>
<td>Names the owner of a view</td>
</tr>
</tbody>
</table>
### ALTER ALIAS statement

The ALTER ALIAS statement specifies changes to the definition of an alias.

**Figure 142: ALTER ALIAS statement**

```
ALTER ALIAS

<table>
<thead>
<tr>
<th>OWNER</th>
<th>aliasOwner 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>aliasName 2</td>
</tr>
<tr>
<td>TABLE</td>
<td></td>
</tr>
<tr>
<td>COMMENT</td>
<td>'string'</td>
</tr>
<tr>
<td>NOCOMMENT</td>
<td></td>
</tr>
<tr>
<td>LABEL</td>
<td>'string'</td>
</tr>
<tr>
<td>NOLABEL</td>
<td></td>
</tr>
</tbody>
</table>
```

**ALTER ALIAS aliasOwner 1.aliasName 1**

This option specifies the fully qualified name of the alias to be modified.

**OWNER aliasOwner 2**

This option specifies the new owner of the alias.

**NAME aliasName 2**

This option specifies the new name of the alias.

**TABLE**

This option specifies the fully qualified name or name of the table or view that the alias references. Table 28 on page 380 lists valid values for the TABLE parameter.

---

<table>
<thead>
<tr>
<th>Variable</th>
<th>Identifier type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>volumeSerialNumber</td>
<td>NA</td>
<td>Names a volume&lt;br&gt;The ID can be 1 to 6 characters in length.</td>
</tr>
</tbody>
</table>
Table 28: TABLE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>locationID.tableOwner.tableName</td>
<td>Specifies the fully qualified name of the table that the alias references.</td>
</tr>
<tr>
<td>tableOwner.tableName</td>
<td>Specifies the name of the table that the alias references.</td>
</tr>
<tr>
<td>locationID.viewOwner.viewName</td>
<td>Specifies the fully qualified name of the view that the alias references.</td>
</tr>
<tr>
<td>viewOwner.viewName</td>
<td>Specifies the name of a view that the alias references.</td>
</tr>
</tbody>
</table>

**COMMENT 'string'**

This option specifies a comment for the alias.

**NOCOMMENT**

This option removes an existing comment.

**LABEL 'string'**

This option specifies a label for the alias.

**NOLABEL**

This option removes an existing label.

**ALTER AUXILIARY TABLE statement**

The ALTER AUXILIARY TABLE statement defines the changes to an auxiliary table.

**Figure 143: ALTER AUXILIARY TABLE statement**
ALTER AUXILIARY TABLE tableOwner .tableName

This option specifies the fully qualified name of the auxiliary table to be changed.

OWNER tableOwner

This option specifies the owner of the auxiliary table.

NAME tableName

This option specifies the name of the auxiliary table.

TABLESPACE tableSpaceName

This option specifies the table space for the auxiliary table.

COMMENT 'string'

This option specifies a comment for the auxiliary table.

LABEL 'string'

This option specifies a label for the auxiliary table.

**ALTER CHECK statement**

The ALTER CHECK statement defines the changes to a check constraint.

**Figure 144: ALTER CHECK statement**

ALTERT CHECK tableOwner .tableName .checkConstraintName

This option specifies the fully qualified name of the check constraint to be changed.
NAME `checkConstraintName` 2

This option specifies the new name of the check constraint.

PATH (`schemaList`)

This option specifies the path for the check constraint.

CHECK (`constraintText`)

This option specifies the check constraint text.

**ALTER UNIQUE CONSTRAINT statement**

The ALTER UNIQUE CONSTRAINT statement defines changes to a unique constraint.

**Figure 145: ALTER UNIQUE CONSTRAINT statement**

```
ALTER UNIQUE CONSTRAINT
```

This option specifies the fully qualified name of the unique constraint to be changed.

NAME `uniqueConstraintName` 2

This option specifies the new name of the unique constraint.

KEYCOLUMNS (`columnName`, ...)

This option specifies the replacement columns that define the unique constraint.
ALTER DATABASE statement

The ALTER DATABASE statement specifies changes to a database definition.

Figure 146: ALTER DATABASE statement

ALTER DATABASE

ALTER DATABASE databaseName 1

NAME databaseName 2

OWNER authorizationID

STOGROUP storageGroupName

BUFFERPOOL bufferPoolName

CCSID

ASCII

EBCDIC

UNICODE

<DEFLT>

INDEXBP bufferPoolName

<DEFLT>

ALTER DATABASE databaseName 1

This option specifies the name of the database to be modified.

NAME databaseName 2

This option specifies the new name of the database.

OWNER authorizationID

This option specifies the new owner of the database.

STOGROUP

This option specifies the default storage group for the database. If the STOGROUP parameter is not specified, the default DB2 SYSDEFLT storage group is used. Table 29 on page 384 lists valid values for the STOGROUP parameter.
### Table 29: STOGROUP parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>storageGroupName</td>
<td>Specifies the name of the storage group.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

**BUFFERPOOL**

This option specifies the new default buffer pool for the database for the creation of the table space. If the BUFFERPOOL parameter is not specified, the DB2 default value is used. Table 30 on page 384 lists valid values for the BUFFERPOOL parameter.

### Table 30: BUFFERPOOL parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufferPoolName</td>
<td>Specifies the name of the buffer pool.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

**CCSID**

This option specifies the encoding scheme of the database. Table 31 on page 384 lists valid values for the CCSID parameter.

### Table 31: CCSID parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>Specifies ASCII encoding.</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>Specifies EBCDIC encoding.</td>
</tr>
<tr>
<td>UNICODE</td>
<td>Specifies Unicode encoding.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. The default encoding scheme is determined in the installation options module with the CCSID keyword.</td>
</tr>
</tbody>
</table>

**INDEXBP**

This option specifies the default buffer pool for the database for the creation of the index. Table 32 on page 385 lists valid values for the CCSID parameter. If the INDEXBP parameter is not specified, the DB2 default value is used.
Table 32: INDEXBP parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufferPoolName</td>
<td>Specifies the name of the buffer pool.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

**ALTER FOREIGN KEY statement**

The ALTER FOREIGN KEY statement defines changes to a foreign key.

**Figure 147: ALTER FOREIGN KEY statement**

**ALTER FOREIGN KEY**

```
ALTER FOREIGN KEY  
```

- **NAME** `constraintName 2`
- **REFERENCE TB**
  - `tableOwner 2 . tableName 2`
- **KEYCOLUMNS**
  - `columnName`
- **REFCOLUMNS**
  - `columnName`
- **ON DELETE**
  - **CASCADE**
  - **RESTRICT**
  - **SET NULL**
  - **NO ACTION**
- **ENFORCED**
  - **YES**
  - **NO**

**ALTER FOREIGN KEY `tableOwner 1 . tableName 1 . constraintName 1`**

This option specifies the fully qualified name of the foreign key to be changed.

**NAME `constraintName 2`**

This option specifies the new name of the foreign key. If the NAME parameter is not specified, DB2 automatically generates the name of the foreign key.
REFERENCETB `tableOwner 2`, `tableName 2`  

This option specifies the name of the table to which the foreign key references.

KEYCOLUMNS (`columnName, ...`)  

This option specifies the columns that are included in the definition of the foreign key.

REFCOLUMNS (`columnName, ...`)  

This option specifies the names of the parent key columns.

---

**Note**  
REFCOLUMNS should only be specified when the foreign key constraint name is not specified. To change the parent column list, alter the unique index that the foreign key uses.

---

ON DELETE  

This option specifies the new rule that determines the action to take when a row of the parent table for the foreign key is deleted. Table 33 on page 386 lists valid values for the ON DELETE parameter.

---

**Note**  
You should not build ALTER FOREIGN KEY statement when a foreign key changes  

- From referencing a primary key to referencing a unique key  
- To a new unique key  
- From a unique key reference to a primary key reference

In these instances, you should build a DROP FOREIGN KEY statement and a CREATE FOREIGN KEY instead of an ALTER FOREIGN KEY statement.

---

**Table 33: ON DELETE parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASCADE</td>
<td>Specifies a delete rule of CASCADE.</td>
</tr>
<tr>
<td>RESTRICT</td>
<td>Specifies a delete rule of RESTRICT if the value of the CURRENT RULES special register is DB2.</td>
</tr>
<tr>
<td>SET NULL</td>
<td>Specifies a delete rule of SET NULL if a column of the foreign key allows null values.</td>
</tr>
</tbody>
</table>
NO ACTION

Specifies a delete rule of NO ACTION if the value of the CURRENT RULES special register is SQL.

ENFORCED

This option specifies whether DB2 enforces the referential constraint. Table 34 on page 387 lists valid values for the ENFORCED parameter.

Table 34: ENFORCED parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies that DB2 enforces the referential constraint.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies that DB2 does not enforce the referential constraint.</td>
</tr>
</tbody>
</table>

**ALTER INDEX statement**

The ALTER INDEX statement defines changes to an index.
**Figure 148: ALTER INDEX statement**

### ALTER INDEX

```
ALTER INDEX indexOwner_1 . indexName_1
```

This option specifies the fully qualified name of the index to be changed.

**OWNER indexOwner_2**

This option specifies the new owner of the index.
NAME indexName

This option specifies the new name of the index.

TABLE tableOwner.tableName

This option specifies the table on which the index is created. If the only change to the index is the table name that it references, CHANGE MANAGER does not generate CDL. If you change the index to reference a different table, CHANGE MANAGER generates DROP INDEX and CREATE INDEX CDL statements.

KEYCOLUMNS (columnName order, ...)

This option specifies the column list that defines the key for this index. Table 35 on page 389 lists valid values for the order variable in the KEYCOLUMNS parameter.

Table 35: Order values for the KEYCOLUMNS parameter

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASC (default value)</td>
<td>Places the key columns in ascending order. The key columns for indexes that are defined on an XML column and indexes on an expression can be placed only in ascending order.</td>
</tr>
<tr>
<td>DESC</td>
<td>Places the key columns in descending order.</td>
</tr>
<tr>
<td>RANDOM</td>
<td>Places the key columns in a random order.</td>
</tr>
<tr>
<td>INCLUDE</td>
<td>Specifies an additional column to be appended to the index key columns for a unique index.</td>
</tr>
</tbody>
</table>

KEYTEXT (expression)

This option specifies the parts of the XML documents in an XML column that are indexed (XMLPATTERN).

XMLdataType

For an XML index, this option specifies the SQL data type. The XMLdataType can be one of the following types:

- VARCHAR(integer)
- DECFLOAT
- DATE
- TIMESTAMP(12)
PERIOD BUSINESS_TIME

This option indicates whether to add starting and ending columns of the BUSINESS_TIME period to the index. The starting and ending columns are defined in the table on which the index is created. Table 36 on page 390 lists valid values for the PERIOD BUSINESS_TIME parameter.

Table 36: PERIOD BUSINESS_TIME parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to add starting and ending columns of the BUSINESS_TIME period.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies not to add starting and ending columns of the BUSINESS_TIME period.</td>
</tr>
</tbody>
</table>

UNIQUE

This option indicates whether index values must be unique. Table 37 on page 390 lists valid values for the UNIQUE parameter.

Table 37: UNIQUE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Defines the index as unique.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies to allow duplicates.</td>
</tr>
<tr>
<td>NOTNULL</td>
<td>Defines the index as unique and specifies that null values are not equal.</td>
</tr>
</tbody>
</table>

COMMENT 'string'

This option specifies the comment for the index.

NOCOMMENT

This option removes the comment for the index.

NUMPARTS number

This option specifies the new number of partitions for the index. NUMPARTS=0 specifies a nonpartitioned index.

ALTER PART partitionNumber indexPartitionParameters

For a partitioned index, this option alters partitionNumber according to the parameters of indexPartitionParameters (see “indexPartitionParameters” on page 392). To alter parameters for a nonpartitioned index, the
partitionNumber value must be zero. For example, the following example changes the value for the PCTFREE attribute for a nonpartitioned index to 20.

```
ALTER INDEX ...
ALTER PART 0 PCTFREE 20
```

**ADD PART BEFORE partitionNumber indexPartitionParameters**

For a partitioned index, this option adds a new partitionNumber according to the parameters of indexPartitionParameters (see “indexPartitionParameters” on page 392).

**ADD PART AFTER partitionNumber indexPartitionParameters**

For a partitioned index, this option adds a new partitionNumber according to the parameters of indexPartitionParameters (see “indexPartitionParameters” on page 392).

**DROP PART number**

For a partitioned index, this option drops the specified partition.

**indexOptions**

This option provides additional options for indexes (see “indexOptions” on page 394).
**indexPartitionParameters**

indexPartitionParameters includes the following parameters.

**Figure 149: indexPartitionParameters**

VCAT *catalogName*

This option specifies the volume catalog to be used for the index.

**STOGROUP**

This option specifies the new default storage group for the index. Table 38 on page 392 lists valid values for the STOGROUP parameter.

**Table 38: STOGROUP parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>storageGroupName</td>
<td>Specifies the name of the storage group.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>
**PRIQTY number**

This option specifies the primary space that is allocated for the DB2-defined data set.

**SECQTY number**

This option specifies the secondary space that is allocated for the DB2-defined data set.

**ERASE**

This option indicates whether to erase the data sets when they are deleted by a statement that drops the index. Table 39 on page 393 lists valid values for the ERASE parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to erase the data sets.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies not to erase the data sets.</td>
</tr>
</tbody>
</table>

**FREEPAGE number**

This option specifies that a free page is added after every number of pages upon index creation or reorganization.

**PCTFREE number**

This option specifies the percentage of free space left on each page of the index upon creation or reorganization.

**GBPCACHE**

This option specifies which index pages are cached to the group buffer pool. Table 40 on page 393 lists valid values for the GBPCACHE parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANGED</td>
<td>Specifies to cache updated index pages to the buffer pool.</td>
</tr>
<tr>
<td>ALL</td>
<td>Specifies to cache all index pages as they are read.</td>
</tr>
<tr>
<td>NONE</td>
<td>Specifies not to cache index pages to the buffer pool.</td>
</tr>
</tbody>
</table>
VALUES (constant, ...)

This option specifies the replacement key limit list for this part of a partitioned index.

*indexOptions*

*indexOptions* includes the following parameters.

**Figure 150: indexOptions**

**CLUSTER**

This option indicates whether the index is defined as a clustering index. Table 41 on page 394 lists valid values for the CLUSTER parameter.

**Table 41: CLUSTER parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to use the index as the clustering index for the table.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies not to use the index as the clustering index for the table.</td>
</tr>
</tbody>
</table>

**PADDED**

This option indicates how columns that are defined as VARCHAR are stored in the index. Table 42 on page 395 lists valid values for the PADDED parameter.

**Table 42: PADDED parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that the columns should not be padded to the length of the index.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that the columns should be padded.</td>
</tr>
</tbody>
</table>

**COMPRESS**

This option indicates whether index data is compressed. Table 43 on page 395 lists valid values for the COMPRESS parameter.

**Table 43: COMPRESS parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to use index compression</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to use index compression</td>
</tr>
</tbody>
</table>

**DEFINE**

This option specifies when the data sets for the index are created. Table 44 on page 395 lists valid values for the DEFINE parameter.

**Table 44: DEFINE parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES (default value)</td>
<td>Indicates that DB2 creates the data sets when it alters the index.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that DB2 creates the data sets when a row is inserted into a table.</td>
</tr>
</tbody>
</table>

**PARTITIONED**

This option indicates whether the index is partitioned. Table 45 on page 396 lists valid values for the PARTITIONED parameter.
Table 45: PARTITIONED parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that the index is partitioned.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that the index is not partitioned.</td>
</tr>
</tbody>
</table>

**BUFFERPOOL**

This option specifies the new default buffer pool for the index. Table 46 on page 396 lists valid values for the BUFFERPOOL parameter.

Table 46: BUFFERPOOL parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufferPoolName</td>
<td>Specifies the name of the buffer pool.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

**CLOSE**

This option indicates whether the index can be closed when the index is not being used. Table 47 on page 396 lists valid values for the CLOSE parameter.

Table 47: CLOSE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES(default value)</td>
<td>Indicates that the index can be closed.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that the index cannot be closed.</td>
</tr>
</tbody>
</table>

**PIECESIZE integer keyword**

This option specifies the maximum PIECESIZE for a nonpartitioned index. The default PIECESIZE is 2 G (2 GB) for indexes backed by non-LARGE table spaces and 4 G (4 GB) for indexes backed by LARGE table spaces. The subsequent keyword (K, G, or M) indicates the value of the specific integer (Table 48 on page 396).

Table 48: PIECESIZE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>K</td>
<td>Specifies the PIECESIZE in kilobytes.</td>
</tr>
<tr>
<td>M</td>
<td>Specifies the PIECESIZE in megabytes.</td>
</tr>
<tr>
<td>G</td>
<td>Specifies the PIECESIZE in gigabytes.</td>
</tr>
</tbody>
</table>
COPY

This option indicates whether to use a utility to create an image copy of the index. Table 49 on page 397 lists valid values for the COPY parameter.

Table 49: COPY parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO (default value)</td>
<td>Indicates not to use a utility to create an image copy.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates to use a utility to create an image copy.</td>
</tr>
</tbody>
</table>

ALTER PROCEDURE statement

The ALTER PROCEDURE statement defines changes to a stored procedure.
Figure 151: ALTER PROCEDURE statement (page 1)

```
ALTER PROCEDURE
   procedureOwner - . - procedureSchema - . - procedureName

   VERSION - versionNumber

   PARAMETERS - ( ordinality, parmUsage, parmName, dataType, subType, locator - )

   ACTIVE - NO

   YES

   APPLCOMPAT
      V10R1
      V11R1
      V12R1Mnnn
      <DEFLT>
      <BLANK>

   APPLICATION_ENCODING_SCHEME
      ASCII
      EBCDIC
      UNICODE
      <DEFLT>

   ARCHIVE SENSITIVE - NO

   YES

   ASUTIME n

   BUSINESS TIME SENSITIVE - NO

   YES

   COLLID
      <NONE>

   collectionID

   COMMENT
      remarks

   <NONE>

   COMMIT_ON_RETURN
      NO

      YES

   COMMIT_ON_RETURN
      AUTONOMOUS

   CONCURRENT_ACCESS_RESOLUTION - <DEFLT>

   COMMIT
      WAIT

   USE

   CURRENT_DATA - NO

   YES

   alterProcedure page 2
```
ALTER PROCEDURE procedureOwner.procedureSchema.procedureName

This option specifies the fully qualified name of the procedure.

VERSION versionNumber

This option specifies the version of the procedure.

PARAMETERS (ordinality, parmUsage, parmName, dataType, subType, locator)

This option specifies the parameters for a procedure. Table 50 on page 399 lists valid values for the procedure parameters.

### Table 50: Procedure parameters values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ordinality</td>
<td>Specifies the ordinal number of the parameter.</td>
</tr>
<tr>
<td>parmUsage</td>
<td>Specifies whether the parameter is used as input, output, or both.</td>
</tr>
<tr>
<td>parmName</td>
<td>Specifies the name of the parameter.</td>
</tr>
<tr>
<td>dataType</td>
<td>Specifies the data type of the parameter.</td>
</tr>
<tr>
<td>subType</td>
<td>Specifies the subtype of the distinct data type.</td>
</tr>
<tr>
<td>locator</td>
<td>Indicates whether a locator to a value or the value itself is returned.</td>
</tr>
</tbody>
</table>

ACTIVE

This option indicates whether to activate the version of the procedure. Table 51 on page 399 lists valid values for the ACTIVE keyword.

### Table 51: ACTIVE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that the version should not be recognized as the active version.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to activate the version.</td>
</tr>
</tbody>
</table>

APPLCOMPAT

This option indicates the package compatibility level for SQL.

Table 52 on page 400 lists valid values for the APPLCOMPAT keyword.
Table 52: APPLCOMPAT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>V10R1</td>
<td>Specifies that the SQL statements in the package are compatible with DB2 Version 10 Release 1.</td>
</tr>
<tr>
<td>V11R1</td>
<td>Specifies that the SQL statements in the package are compatible with DB2 Version 11 Release 1.</td>
</tr>
<tr>
<td>V12R1Mnnn</td>
<td>Specifies that the SQL statements in the package are compatible with DB2 Version 12 Release 1 Modification nnn.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
<tr>
<td>&lt;BLANK&gt;</td>
<td>Indicates the old value of the keyword when objects have been catmantained from an earlier version of DB2.</td>
</tr>
</tbody>
</table>

**APPLICATION_ENCODING_SCHEME**

This option specifies the encoding scheme for the procedure. Table 53 on page 400 lists valid values for the APPLICATION_ENCODING_SCHEME keyword.

Table 53: APPLICATION_ENCODING_SCHEME keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>Specifies ASCII encoding.</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>Specifies EBCDIC encoding.</td>
</tr>
<tr>
<td>UNICODE</td>
<td>Specifies Unicode encoding.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

**ARCHIVE SENSITIVE**

This option specifies whether references to archive-enabled tables in SQL statements are affected by the value of the SYSIBMADM.GET_ARCHIVE global variable. Table 54 on page 400 lists valid values for the ARCHIVE SENSITIVE keyword.

Table 54: ARCHIVE SENSITIVE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that references are not affected.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies that references are affected.</td>
</tr>
</tbody>
</table>
BUSINESS_TIME SENSITIVE

This option specifies whether references to application-period temporal tables in both static and dynamic SQL statements are affected by the value of the CURRENT TEMPORAL BUSINESS_TIME special register. Table 55 on page 401 lists valid values for the BUSINESS_TIME SENSITIVE keyword.

Table 55: BUSINESS_TIME SENSITIVE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that references are not affected.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies that references are affected.</td>
</tr>
</tbody>
</table>

ASUTIME \( n \)

This option specifies the number of CPU service units for the procedure. A value of 0 indicates that no limit exists on the number of units.

COLLID

This option specifies the name of the package collection for the procedure. (The name of this collection is stored in the SYSIBM.SYSROUTINES table.) Table 56 on page 401 lists valid values for the COLLID keyword.

Table 56: COLLID keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;NONE&gt;</td>
<td>If a blank value is specified for the collection, specifies NO COLLID.</td>
</tr>
<tr>
<td>collectionID</td>
<td>Specifies the name of the package collection for the procedure.</td>
</tr>
</tbody>
</table>

COMMENT

This option specifies the comments for a procedure. Table 57 on page 401 lists valid values for the COMMENT keyword.

Table 57: COMMENT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>remarks</td>
<td>Specifies a comment for the procedure.</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies no comments for the procedure.</td>
</tr>
</tbody>
</table>

COMMIT_ON_RETURN NO

This option specifies that DB2 should continue the unit of work
COMMIT_ON_RETURN YES

This option specifies that DB2 should commit the unit of work immediately.

AUTONOMOUS

For native SQL procedures, this option specifies that DB2 should execute the SQL procedure in a unit of work that is independent from the calling application.

CONCURRENT_ACCESS_RESOLUTION

This option specifies the concurrent access resolution option to use for statements in a package. Table 58 on page 402 lists valid values for the CONCURRENT_ACCESS_RESOLUTION keyword.

Table 58: CONCURRENT_ACCESS_RESOLUTION keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
<tr>
<td>WAIT</td>
<td>Specifies the WAITFOROUTCOME option.</td>
</tr>
<tr>
<td>USE</td>
<td>Specifies the USECURRENTLYCOMMITTED option.</td>
</tr>
</tbody>
</table>

CURRENT_DATA

This option indicates whether to require data currency for cursors when the isolation level of cursor stability is in effect. Table 59 on page 402 lists valid values for the CURRENT_DATA keyword.

Table 59: CURRENT_DATA keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies to not require data currency.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to require data currency.</td>
</tr>
</tbody>
</table>
### Figure 152: ALTER PROCEDURE statement (page 2)

#### alterProcedure page 2

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE_FORMAT</td>
<td>&lt;DEFLT&gt;</td>
</tr>
<tr>
<td></td>
<td>EUR</td>
</tr>
<tr>
<td></td>
<td>ISO</td>
</tr>
<tr>
<td></td>
<td>JIS</td>
</tr>
<tr>
<td></td>
<td>LOCAL</td>
</tr>
<tr>
<td></td>
<td>USA</td>
</tr>
<tr>
<td>DBINFO</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>YES</td>
</tr>
<tr>
<td>DEBUG_MODE</td>
<td>ALLOW</td>
</tr>
<tr>
<td></td>
<td>DISALLOW</td>
</tr>
<tr>
<td></td>
<td>DISABLE</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt;</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>(x, y)</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt;</td>
</tr>
<tr>
<td>DEFER_PREPARE</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt;</td>
</tr>
<tr>
<td>DEGREE</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ANY</td>
</tr>
<tr>
<td>DETERMINISTIC</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>YES</td>
</tr>
<tr>
<td>DYNAMICRULES</td>
<td>RUN</td>
</tr>
<tr>
<td></td>
<td>BIND</td>
</tr>
<tr>
<td></td>
<td>DEFINEBIND</td>
</tr>
<tr>
<td></td>
<td>DEFINERUN</td>
</tr>
<tr>
<td></td>
<td>INVOKEBIND</td>
</tr>
<tr>
<td></td>
<td>INVOKERUN</td>
</tr>
<tr>
<td>EXPLAIN</td>
<td>NO</td>
</tr>
<tr>
<td></td>
<td>YES</td>
</tr>
</tbody>
</table>

---

*Chapter 6  Change Definition Language  403*
DATE_FORMAT

This option specifies the format for the date for the procedure. Table 60 on page 404 lists valid values for the DATE_FORMAT keyword.

Table 60: DATE_FORMAT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
<tr>
<td>EUR</td>
<td>Specifies the IBM European standard (dd.mm.yyyy)</td>
</tr>
<tr>
<td>ISO</td>
<td>Specifies the International Standards Organization format (yyyy-mm-dd)</td>
</tr>
<tr>
<td>JIS</td>
<td>Specifies the Japanese industrial standard Christian era format (yyyy-mm-dd)</td>
</tr>
<tr>
<td>LOCAL</td>
<td>Specifies the installation-defined format</td>
</tr>
<tr>
<td>USA</td>
<td>Specifies the IBM USA standard (mm/dd/yyyy)</td>
</tr>
</tbody>
</table>

DBINFO

This option indicates whether to pass the DBINFO structure to the stored procedure. Table 61 on page 404 lists valid values for the DBINFO keyword.

Table 61: DBINFO keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to pass the DBINFO structure to the stored procedure.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to pass the DBINFO structure to the stored procedure.</td>
</tr>
</tbody>
</table>

DEBUG_MODE

This option specifies whether the procedure is allowed for debugging. Table 62 on page 404 lists valid values for the DEBUG_MODE keyword.

Table 62: DEBUG_MODE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLOW</td>
<td>Specifies to allow debugging.</td>
</tr>
<tr>
<td>DISALLOW</td>
<td>Specifies not to allow debugging.</td>
</tr>
<tr>
<td>DISABLE</td>
<td>Specifies to disable debugging.</td>
</tr>
</tbody>
</table>
### DECIMAL

This option specifies the rules for a decimal operation (DEC31 or DEC15). Table 63 on page 405 lists valid values for the DECIMAL keyword.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x, y)</td>
<td>Specifies the decimal value (x) and the scale (y)</td>
</tr>
<tr>
<td></td>
<td>The decimal value is either 31 or 15. Values for scale are 1 through 9.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

### DEFER_PREPARE

This option indicates whether to immediately prepare dynamic SQL statements or to defer preparation. Table 64 on page 405 lists valid values for the DEFER_PREPARE keyword.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to defer the preparation of dynamic SQL statements.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies to immediately prepare dynamic SQL statements.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifieds a default value or to inherit the option from the plan.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

### DEGREE

This option specifies whether to attempt to run a query using parallel processing. Table 65 on page 405 lists valid values for the DEGREE keyword.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Specifies to prohibit parallel processing.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>ANY</td>
<td>Specifies to allow parallel processing.</td>
</tr>
</tbody>
</table>

**DETERMINISTIC**

This option indicates whether to use indeterminate or deterministic results for a procedure. Table 66 on page 406 lists valid values for the DETERMINISTIC keyword.

**Table 66: DETERMINISTIC keyword values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies to use indeterminate results.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to use deterministic results.</td>
</tr>
</tbody>
</table>

**DYNAMICRULES**

This option specifies the values for the dynamic SQL attributes. Table 67 on page 406 lists valid values for the DYNAMICRULES keyword.

**Table 67: DYNAMICRULES keyword values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUN</td>
<td>Specifies to process dynamic SQL statements using standard attribute values.</td>
</tr>
<tr>
<td>BIND</td>
<td>Specifies to process dynamic SQL statements using bind behavior.</td>
</tr>
<tr>
<td>DEFINEBIND</td>
<td>Specifies to process dynamic SQL statements using define behavior or bind behavior.</td>
</tr>
<tr>
<td>DEFINERUN</td>
<td>Specifies to process dynamic SQL statements using define behavior or run behavior.</td>
</tr>
<tr>
<td>INVOKEBIND</td>
<td>Specifies to process dynamic SQL statements using invoke behavior or bind behavior.</td>
</tr>
<tr>
<td>INVOKE_RUN</td>
<td>Specifies to process dynamic SQL statements using invoke behavior or run behavior.</td>
</tr>
</tbody>
</table>

**EXPLAIN**

This option indicates whether to add information about the package’s statements to the owner of the PLAN_TABLE table. Table 68 on page 407 lists valid values for the EXPLAIN keyword.
Table 68: EXPLAIN keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to add information.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to add information.</td>
</tr>
</tbody>
</table>
Figure 153: ALTER PROCEDURE statement (page 3)

```
alterProcedure (page 3)

- EXTERNAL_NAME  externalName  JAVAExternalName
- IMMEDIATE_WRITE  NO  YES
- ISOLATION_LEVEL  CS  RR  RS  UR
- KEEP_DYNAMIC  NO  YES
- LANGUAGE  ASSEMBLE
  - C
  - COBOL
  - PLI
  - REXX
  - SQL
  - JAVA
- MAXIMUM_FAILURES  -1
  - 0
  - n
- OPTINT  'value'
  - <NONE>
- OWNER  authorizationID
- PACKAGE_OWNER  owner
  - <NONE>
- PACKAGE_PATH  <NONE>
  - 'value1,value2,value3'
- PARAMETER_CCSID  ASCII
  - EBCDIC
  - UNICODE
  - <DEFLT>
- PARAMETER_STYLE  SQL
  - GENERAL_NO_NULLS
  - GENERAL_WITH_NULLS
  - JAVA
```

alterProcedure page 4
EXTERNAL_NAME externalName JAVAExternalName

This option specifies the module that DB2 should load to execute the procedure. For parameters for a JAVAExternalName, see Figure 154 on page 411.

IMMEDIATE_WRITE

This option indicates whether updates that are made to dependent page sets for group buffer pools should be immediately written. Table 69 on page 409 lists valid values for the IMMEDIATE_WRITE keyword.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies normal write activity.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to immediately write updated pages.</td>
</tr>
</tbody>
</table>

ISOLATION_LEVEL

This option specifies the isolation level. Table 70 on page 409 lists valid values for the ISOLATION_LEVEL keyword.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>Specifies cursor stability.</td>
</tr>
<tr>
<td>RR</td>
<td>Specifies repeatable read.</td>
</tr>
<tr>
<td>RS</td>
<td>Specifies read stability.</td>
</tr>
<tr>
<td>UR</td>
<td>Specifies uncommitted read.</td>
</tr>
</tbody>
</table>

KEEP_DYNAMIC

This option indicates whether dynamic SQL statements are purged at each commit point. Table 71 on page 409 lists valid values for the KEEP_DYNAMIC keyword.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies to purge dynamic SQL statements.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to keep dynamic SQL statements.</td>
</tr>
</tbody>
</table>
LANGUAGE

This option specifies the implementation language of the procedure. Table 72 on page 410 lists valid values for the LANGUAGE keyword.

### Table 72: LANGUAGE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSEMBLE</td>
<td>Specifies assembler.</td>
</tr>
<tr>
<td>C</td>
<td>Specifies C.</td>
</tr>
<tr>
<td>COBOL</td>
<td>Specifies COBOL.</td>
</tr>
<tr>
<td>PLI</td>
<td>Specifies PL/1.</td>
</tr>
<tr>
<td>REXX</td>
<td>Specifies REXX.</td>
</tr>
<tr>
<td>SQL</td>
<td>Specifies SQL.</td>
</tr>
<tr>
<td>JAVA</td>
<td>Specifies JAVA.</td>
</tr>
</tbody>
</table>

MAXIMUM_FAILURES

This option specifies the number of allowable failures for the procedure. Table 73 on page 410 lists valid values for the MAXIMUM_FAILURES keyword.

### Table 73: MAXIMUM_FAILURES keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>Specifies to use the DB2 installation parameter (STORMXAB).</td>
</tr>
<tr>
<td>0</td>
<td>Specifies to never stop the procedure.</td>
</tr>
<tr>
<td>n</td>
<td>Specifies the number of failures. Values are 1 through 32767.</td>
</tr>
</tbody>
</table>

OPTHINT

This option specifies the value of the query optimization hint bind option. Table 74 on page 410 lists valid values for the OPTHINT keyword.

### Table 74: OPTHINT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>'value'</td>
<td>Specifies a character string to be used by the optimizer when searching for input to DB2</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies that DB2 should not use optimization hints for SQL statements</td>
</tr>
</tbody>
</table>
OWNER authorizationID

This option specifies the owner of the procedure. The OWNER parameter is included in the CDL statement only if the OVERRIDE(SPOWNER) keyword is in the ALUIN input stream.

When the procedure is a JAVA procedure, JAVAExternalName uses the following parameters for an external name.

Figure 154: JAVAExternalName

JAVAExternalName

<table>
<thead>
<tr>
<th>JAR_SCHEMA ——— schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>JAR_ID ——— ID</td>
</tr>
<tr>
<td>EXTERNAL_NAME ——— name</td>
</tr>
<tr>
<td>JAVASIGNATURE ——— signature</td>
</tr>
</tbody>
</table>

JAR_SCHEMA schema

This option specifies the schema of the JAR file.

JAR_ID name

This option specifies the name of the JAR file.

EXTERNAL_NAME name

This option specifies the package.class.method that DB2 should load to execute the JAVA procedure.

JAVASIGNATURE signature

This option specifies the signature of the JAR file.

PACKAGE_OWNER

This option specifies the owner of the package. Table 75 on page 412 lists valid values for the PACKAGE_OWNER keyword.
Table 75: PACKAGE_OWNER keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>Specifies the authorization ID of the owner of the package.</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies no owner for the package.</td>
</tr>
</tbody>
</table>

**PACKAGE_PATH**

This option specifies the package path to use when the procedure executes. Table 76 on page 412 lists valid values for the PACKAGE_PATH keyword.

Table 76: PACKAGE_PATH keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies the package path of the program that invoked the procedure.</td>
</tr>
<tr>
<td>'value1, value2, value3'</td>
<td>Specifies the package path.</td>
</tr>
</tbody>
</table>

**PARAMETER_CCSID**

This option specifies the encoding scheme for the parameters for the procedure. Table 77 on page 412 lists valid values for the PARAMETER_CCSID keyword.

Table 77: PARAMETER_CCSID keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>Specifies ASCII encoding.</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>Specifies EBCDIC encoding.</td>
</tr>
<tr>
<td>UNICODE</td>
<td>Specifies Unicode encoding.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

**PARAMETER_STYLE**

This option specifies the convention used to pass parameters to and return values from the procedure. Table 78 on page 413 lists valid values for the PARAMETER_STYLE keyword.
Table 78: PARAMETER_STYLE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL</td>
<td>Specifies to pass parameters according to the DB2SQL standard convention.</td>
</tr>
<tr>
<td>GENERAL_NO_NULLS</td>
<td>Specifies to pass parameters according to the GENERAL standard convention.</td>
</tr>
<tr>
<td>GENERAL_WITH_NULLS</td>
<td>Specifies to pass parameters according to the GENERAL WITH NULLS standard convention.</td>
</tr>
<tr>
<td>JAVA</td>
<td>Specifies to pass parameters according to the JAVA and SQLJ standard conventions.</td>
</tr>
</tbody>
</table>
Figure 155: ALTER PROCEDURE statement (page 4)

```plaintext
alterProcedure (page 4)
```

```
PARAMETER_VARCHAR
   <DEFLT>
      NULLTERM
      STRUCTURE

PROGRAM_TYPE
   SUB
   MAIN

QUALIFIER
   qualifier
   <DEFLT>

RELEASE_AT
   COMMIT
   DEALLOCATE

RESULT_SETS
   value

REOPT
   NONE
   ALWAYS
   ONCE

 ROUNDING
   DEC_ROUND_CEILING
   DEC_ROUND_DOWN
   DEC_ROUND_FLOOR
   DEC_ROUND_HALF_DOWN
   DEC_ROUND_HALF_UP
   DEC_ROUND_HALF_EVEN
   DEC_ROUND_UP
   <DEFLT>

RUN_OPTIONS
   runtimeOptions
   <NONE>

SECURITY
   DB2
   USER
   DEFINER

SPECIAL_REGISTERS
   INHERIT
   DEFAULT

SQL_DATA_ACCESS
   MODIFIES
   READS
   CONTAINS
   NONE
```
PARAMETER_VARCHAR

This option specifies the representation of variable length string parameters. Table 79 on page 415 lists valid values for the PARAMETER_VARCHAR keyword.

Table 79: PARAMETER_VARCHAR keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
<tr>
<td>NULLTERM</td>
<td>Specifies a null-terminated string form.</td>
</tr>
<tr>
<td>STRUCTURE</td>
<td>Specifies a VARCHAR structure form.</td>
</tr>
</tbody>
</table>

PROGRAM_TYPE

This option indicates whether the procedure runs as a main routine or a subroutine. Table 80 on page 415 lists valid values for the PROGRAM_TYPE keyword.

Table 80: PROGRAM_TYPE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUB</td>
<td>Specifies to run the procedure as a subroutine.</td>
</tr>
<tr>
<td>MAIN</td>
<td>Specifies to run the procedure as a main routine.</td>
</tr>
</tbody>
</table>

QUALIFIER

This option specifies the implicit qualifier for the unqualified object names in the SQL statements. Table 81 on page 415 lists valid values for the QUALIFIER keyword.

Table 81: QUALIFIER keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>qualifier</td>
<td>Specifies the qualifier for the unqualified table, view, index, and alias names.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

RELEASE_AT

This option specifies the release resources when the package was last bound or rebound. Table 82 on page 416 lists valid values for the RELEASE_AT keyword.
Table 82: RELEASE_AT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMIT</td>
<td>Specifies the COMMIT value.</td>
</tr>
<tr>
<td>DEALLOCATE</td>
<td>Specifies the DEALLOCATE value.</td>
</tr>
</tbody>
</table>

RESULT_SETS value

This option specifies the number of query result sets that the procedure can return. Values are 0 through 32767.

REOPT

This option specifies whether to have DB2 determine an access path at run time. Table 83 on page 416 lists valid values for the REOPT keyword.

Table 83: REOPT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>Specifies to determine an access path at run time.</td>
</tr>
<tr>
<td>ALWAYS</td>
<td>Specifies to determine the access path at run time each time that the statement runs.</td>
</tr>
<tr>
<td>ONCE</td>
<td>Specifies to determine the access path only once.</td>
</tr>
</tbody>
</table>

ROUNDING

This option specifies the rounding mode when the package is bound. Table 84 on page 416 lists valid values for the ROUNDING keyword.

Table 84: ROUNDING keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEC_ROUND_CEILING</td>
<td>Specifies to round toward infinity</td>
</tr>
<tr>
<td>DEC_ROUND_DOWN</td>
<td>Specifies to round toward zero.</td>
</tr>
<tr>
<td>DEC_ROUND_FLOOR</td>
<td>Specifies to round toward negative infinity.</td>
</tr>
<tr>
<td>DEC_ROUND_HALF_DOWN</td>
<td>Specifies to round to the nearest digit. If the digit is equidistant, round down.</td>
</tr>
<tr>
<td>DEC_ROUND_HALF_EVEN</td>
<td>Specifies to round to the nearest digit. If the digit is equidistant, round to an even digit.</td>
</tr>
<tr>
<td>DEC_ROUND_HALF_UP</td>
<td>Specifies to round to the nearest digit. If the digit is equidistant, round up.</td>
</tr>
<tr>
<td>DEC_ROUND_UP</td>
<td>Specifies to round away from zero.</td>
</tr>
</tbody>
</table>
### RUN_OPTIONS

This option specifies the language environment run-time options for the procedure. Table 85 on page 417 lists valid values for the RUN_OPTIONS keyword.

**Table 85: RUN_OPTIONS keyword values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>runtimeOptions</td>
<td>Specifies the run-time options.</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies to use the default run-time options.</td>
</tr>
</tbody>
</table>

### SECURITY

If the procedure accesses resources that are protected by an external security product, specifies the authorization ID. Table 86 on page 417 lists valid values for the SECURITY keyword.

**Table 86: SECURITY keyword values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB2</td>
<td>Specifies the authorization ID for the WLM procedure address space.</td>
</tr>
<tr>
<td>USER</td>
<td>Specifies the authorization ID of the user who invoked the procedure.</td>
</tr>
<tr>
<td>DEFINER</td>
<td>Specifies the authorization ID of the owner.</td>
</tr>
</tbody>
</table>

### SPECIAL_REGISTERS

This option specifies how special registers are set. Table 87 on page 417 lists valid values for the SPECIAL_REGISTERS keyword.

**Table 87: SPECIAL_REGISTERS keyword values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INHERIT</td>
<td>Specifies that the values of special registers are inherited.</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>Specifies that the values of special registers are initialized to the default values.</td>
</tr>
</tbody>
</table>
SQL_DATA_ACCESS

This option specifies which SQL statements can be executed in the procedure. Table 88 on page 418 lists valid values for the SQL_DATA_ACCESS keyword.

Table 88: SQL_DATA_ACCESS keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODIFIES</td>
<td>Specifies that the procedure can execute only supported SQL statements.</td>
</tr>
<tr>
<td>READS</td>
<td>Specifies that the procedure can execute SQL statements with READS SQL DATA, CONTAINS SQL, or NO SQL.</td>
</tr>
<tr>
<td>CONTAINS</td>
<td>Specifies that the procedure can execute SQL statements with CONTAINS SQL or NO SQL.</td>
</tr>
<tr>
<td>NONE</td>
<td>Specifies that the procedure can execute SQL statements with NO SQL.</td>
</tr>
</tbody>
</table>
This option specifies the SQL path that DB2 uses to resolve unqualified types and names in the procedure. Table 89 on page 420 lists valid values for the SQL_PATH keyword.
Table 89: SQL_PATH keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>Specifies &quot;SYSIBM&quot;, &quot;SYSFUN&quot;, &quot;SYSPROC&quot;, and the value of the QUALIFIER option.</td>
</tr>
<tr>
<td>USER</td>
<td>Specifies the value of the SESSION_USER or USER special register.</td>
</tr>
<tr>
<td>schemaName</td>
<td>Specifies the name of the schema.</td>
</tr>
<tr>
<td>schemaNameList</td>
<td>Specifies a list of schema names, separated by commas.</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies no SQL path.</td>
</tr>
</tbody>
</table>

**STAY_RESIDENT**

This option indicates whether the procedure load module remains resident in memory when the procedure ends. Table 90 on page 420 lists valid values for the STAY_RESIDENT keyword.

Table 90: STAY_RESIDENT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies to delete the load module from memory.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to retain the load module in memory.</td>
</tr>
</tbody>
</table>

**SYSTEM_TIME SENSITIVE**

This option indicates whether references to system-period temporal tables in both static and dynamic SQL statements are affected by the value of the CURRENT TEMPORAL SYSTEM_TIME special register. Table 91 on page 420 lists valid values for the SYSTEM_TIME SENSITIVE keyword.

Table 91: SYSTEM_TIME SENSITIVE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that references are not affected.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies that references are affected.</td>
</tr>
</tbody>
</table>

**TIME_FORMAT**

This option specifies the time format for values. Table 92 on page 421 lists valid values for the TIME_FORMAT keyword.
Table 92: TIME_FORMAT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
<tr>
<td>EUR</td>
<td>Specifies the IBM European standard (hh:mm:ss).</td>
</tr>
<tr>
<td>ISO</td>
<td>Specifies the International Standards Organization format (hh:mm:ss).</td>
</tr>
<tr>
<td>JIS</td>
<td>Specifies the Japanese industrial standard Christian era format (hh:mm:ss).</td>
</tr>
<tr>
<td>LOCAL</td>
<td>Specifies the installation-defined format.</td>
</tr>
<tr>
<td>USA</td>
<td>Specifies the IBM USA standard (hh:mm\ AM or PM).</td>
</tr>
</tbody>
</table>

FOR_UPDATE_CLAUSE

This option specifies whether the FOR UPDATE clause is required or optional for a DECLARE CURSOR statement. Table 93 on page 421 lists valid values for the FOR_UPDATE_CLAUSE keyword.

Table 93: FOR_UPDATE_CLAUSE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQUIRED</td>
<td>Specifies that the clause is required.</td>
</tr>
<tr>
<td>OPTIONAL</td>
<td>Specifies that the clause is optional.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

VALIDATE

This option specifies whether to recheck errors. Table 94 on page 421 lists valid values for the VALIDATE keyword.

Table 94: VALIDATE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUN</td>
<td>Specifies to issue a warning when the CREATE statement succeeds.</td>
</tr>
<tr>
<td>BIND</td>
<td>Specifies to issue an error when the CREATE statement fails.</td>
</tr>
</tbody>
</table>

WLM_ENVIRONMENT

This option specifies the workload manager (WLM) environment in which the procedure runs. Table 95 on page 422 lists valid values for the WLM_ENVIRONMENT keyword.
Table 95: WLM_ENVIRONMENT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
<tr>
<td>WLMName</td>
<td>Specifies the name of the WLM environment.</td>
</tr>
</tbody>
</table>

**WLM_ENV_FOR_NESTED**

For nested procedures, this option indicates which address space DB2 uses. Table 96 on page 422 lists valid values for the WLM_ENV_FOR_NESTED keyword.

Table 96: WLM_ENV_FOR_NESTED keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to use the address space specified in the WLM environment.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies to use an address space other than the one specified in the WLM environment.</td>
</tr>
</tbody>
</table>

**TEXT storedProcedureBody**

This option specifies the statements in the body of the SQL procedure.

**ALTER SEQUENCE statement**

The ALTER SEQUENCE statement defines changes to a sequence definition.
Figure 156: ALTER SEQUENCE statement

ALTER SEQUENCE

This option specifies the sequence to be changed.

SCHEMA schema 2

This option specifies the new schema for the sequence.

NAME sequenceName 2

This option specifies the new name of the sequence.

TYPESCHEMA dataTypeSchema

This option specifies the schema for the sequence data type.
**TYPENAME** `dataTypeName`

This option specifies the name of the sequence data type.

**LENGTH numeric**

For sequences that are defined as a DECIMAL data type, this option specifies the length of the value.

**START numeric**

This option specifies the starting value for the sequence. The default value is the minimum value for an ascending sequence or the maximum value for a descending sequence.

**INCREMENT numeric**

This option specifies the amount of change in the values for the sequence. The value can be any positive or negative integer, including 0 (zero). The default value is 1.

**MINVALUE**

This option specifies the minimum value for a sequence. The minimum value can be equal to or less than the maximum value. Table 97 on page 424 lists valid values for the MINVALUE parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numeric</td>
<td>Specifies the minimum value</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value for the minimum value</td>
</tr>
<tr>
<td></td>
<td>If the sequence is descending, &lt;DEFLT&gt; is the smallest value allowed. If the sequence is ascending, the value is either the starting value or 1 if the starting value is not specified. &lt;DEFLT&gt; is the equivalent of NOMINVALUE.</td>
</tr>
</tbody>
</table>

**MAXVALUE**

This option specifies the maximum value for a sequence. The maximum value can be equal to or greater than the minimum value. Table 98 on page 424 lists valid values for the MAXVALUE parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numeric</td>
<td>Specifies the maximum value</td>
</tr>
</tbody>
</table>
CDL statements

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value for the maximum value</td>
</tr>
<tr>
<td></td>
<td>If the sequence is ascending, &lt;DEFLT&gt; is the largest value allowed. If the</td>
</tr>
<tr>
<td></td>
<td>column is descending, the default value is either the starting value or -1</td>
</tr>
<tr>
<td></td>
<td>if the starting value is not specified. &lt;DEFLT&gt; is the equivalent of NOMAXVALUE.</td>
</tr>
</tbody>
</table>

**CYCLE**

This option indicates whether values should be generated after the maximum or minimum value is reached. Table 99 on page 425 lists valid values for the CYCLE parameter.

**Table 99: CYCLE parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that values should not be generated after the maximum or minimum</td>
</tr>
<tr>
<td></td>
<td>value is reached</td>
</tr>
<tr>
<td></td>
<td>CYCLE NO is the default.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that values should be generated after the maximum or minimum</td>
</tr>
<tr>
<td></td>
<td>value is reached</td>
</tr>
</tbody>
</table>

**CACHE integer**

This option specifies the number of a set of values that are stored for the sequence. A value of 0 (zero) is the equivalent of NO CACHE.

**ORDER**

This option indicates whether values must be generated in the order in which they are requested. Table 100 on page 425 lists valid values for the ORDER parameter.

**Table 100: ORDER parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that values should not be generated in the order in which they are</td>
</tr>
<tr>
<td></td>
<td>requested</td>
</tr>
<tr>
<td></td>
<td>ORDER NO is the default.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that values must be generated in the order in which they are</td>
</tr>
<tr>
<td></td>
<td>requested</td>
</tr>
</tbody>
</table>

**COMMENT 'string'**

This option specifies the comment for the sequence.
**ALTER STOGROUP statement**

The ALTER STOGROUP statement defines changes to a storage group definition.

---

**Figure 157: ALTER STOGROUP statement**

**ALTER STOGROUP**

```
ALTER STOGROUP storageGroupName  
  NAME storageGroupName  
  OWNER authorizationID  
  VOLUMES (volumeSerialNumber, ...)  
  VCAT catalogName  
  DATACLAS name  
  NODATACLAS  
  MGMTCLAS name  
  NOMGMTCLAS  
  STORCLAS name  
  NOSTORCLAS
```

**ALTER STOGROUP storageGroupName 1**

This option specifies the storage group to be changed.

**NAME storageGroupName 2**

This option specifies the name of the storage group.

**OWNER authorizationID**

This option specifies the owner of the storage group.

**VOLUMES (volumeSerialNumber, ...)**

Defines a list of volumes that are included in the storage group.

**VCAT catalogName**

This option specifies the volume catalog for the storage group.

**DATACLAS name**

This option specifies the name of the SMS data class.

**NODATACLAS**

This option removes an existing data class.
MGMTCLAS name

This option specifies the name of the SMS management class.

NOMGMTCLAS

This option removes an existing management class.

STORCLAS name

This option specifies the name of the SMS storage class.

NOSTORCLAS

This option removes an existing storage class.

**ALTER SYNONYM statement**

The ALTER SYNONYM statement defines changes to be made to a DB2 synonym definition.

**Figure 158: ALTER SYNONYM statement**

ALTER SYNONYM

```
ALTER SYNONYM synonymOwner1 . synonymName1
```

OWNER synonymOwner2

This option specifies the new owner of the synonym.

NAME synonymName2

This option specifies the new name of the synonym.

ALTER SYNONYM synonymOwner1 . synonymName1

This option specifies the fully qualified name of the synonym to be changed.
TABLE

This option specifies the new table or view that the synonym references. Table 101 on page 428 lists valid values for the TABLE parameter.

Table 101: TABLE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tableOwner.tableName</code></td>
<td>Specifies the name of a table.</td>
</tr>
<tr>
<td><code>viewOwner.viewName</code></td>
<td>Specifies the name of a view.</td>
</tr>
</tbody>
</table>

ALTER TABLE statement

The ALTER TABLE statement defines the changes to a persistent table or to a global temporary table.
### ALTER TABLE statement

```sql
ALTER TABLE tableOwner_1 . tableName_1
```

This option specifies the fully qualified name of the table to be changed.

### OWNER

```sql
OWNER tableOwner_2
```

This option specifies the new owner of the table.

### NAME

```sql
NAME tableName_2
```

This option specifies the new name of the table.
DATABASE `databaseName`

This option specifies the new database for the table.

TABLESPACE

This option specifies the new table space for the table. Table 102 on page 430 lists valid values for the TABLESPACE parameter.

Table 102: TABLESPACE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tableSpaceName</code></td>
<td>Specifies the name of the table space.</td>
</tr>
<tr>
<td><code>&lt;DEFLT&gt;</code></td>
<td>Specifies a default value for an implicit table space.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;DEFLT&gt;</code> is a reserved keyword.</td>
</tr>
</tbody>
</table>

VALIDPROC `procedureName`

This option defines a new validation procedure.

NOVALIDPROC

This option removes an existing validation procedure.

AUDIT

This option specifies the audit procedure for the table. Table 103 on page 430 lists valid values for the AUDIT parameter.

Table 103: AUDIT parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>Specifies not to perform auditing.</td>
</tr>
<tr>
<td>CHANGES</td>
<td>Specifies to perform auditing when the table is updated.</td>
</tr>
<tr>
<td>ALL</td>
<td>Specifies to perform auditing when any action is performed.</td>
</tr>
</tbody>
</table>

ALTER COLUMN `columnName` 1 (alterColumnSpecification)

Modifies a column according to the specifications in alterColumnSpecification (see “alterColumnSpecification” on page 435).

DROP COLUMN `columnName` 2

This option removes the `columnName` column from the table.
ADD COLUMN \textit{columnName} \textit{3} \textit{position} (\textit{addColumnSpecification})

Adds a new column according to the specifications in \textit{addColumnSpecification} (see “addColumnSpecification” on page 445). Table 104 on page 431 list valid values for the ADD COLUMN parameters.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BEFORE \textit{columnName4}</td>
<td>Specifies that the column will be added before \textit{columnName4}.</td>
</tr>
<tr>
<td>AFTER \textit{columnName4}</td>
<td>Specifies that the column will be added after \textit{columnName4}.</td>
</tr>
<tr>
<td>&lt;LAST&gt;</td>
<td>Indicates that the column will be added to the end of the table.</td>
</tr>
</tbody>
</table>

\textit{partitionParameters}

For a description of the parameters used in \textit{partitionParameters}, see “\textit{partitionParameters}” on page 446.

\textbf{DATA CAPTURE}

This option specifies whether additional information is added to the logging of SQL commands. Table 105 on page 431

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANGES</td>
<td>Specifies to write additional information to the log.</td>
</tr>
<tr>
<td>NONE</td>
<td>Specifies not to include additional information.</td>
</tr>
</tbody>
</table>

\textbf{VOLATILE}

This option specifies whether DB2 should use index access to the table whenever possible. Table 106 on page 431 lists valid values for the VOLATILE parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that SQL access to the table should be based on current statistics.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies that DB2 should use index access whenever possible for SQL operations.</td>
</tr>
</tbody>
</table>

\textbf{DROPRESTRICT}

This option indicates whether to restrict a table from being dropped. Table 107 on page 432 lists valid values for the DROPRESTRICT parameter.
Table 107: DROPRESTRICT parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to restrict the table.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to restrict the table. The parent table space and database of a table with DROPRESTRICT YES cannot be dropped.</td>
</tr>
</tbody>
</table>

**ALTER PERIOD BUSINESS_TIME(columnName 5, columnName 6)**

This option modifies the columns that are used as the beginning and end of the BUSINESS_TIME period. `columnName5` identifies the column that begins the time period. `columnName6` identifies the column that ends the time period.

Table 108 on page 432 lists valid values for the BUSINESS_TIME parameter (valid for DB2 Version 12.1 and later).

Table 108: BUSINESS_TIME parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXCLUSIVE</td>
<td>The beginning value is included in the period, and the ending value is excluded.</td>
</tr>
<tr>
<td>INCLUSIVE</td>
<td>The beginning and ending values are both included in the period.</td>
</tr>
</tbody>
</table>

**ADD PERIOD BUSINESS_TIME(columnName 7, columnName 8)**

This option specifies to add a BUSINESS_TIME period to the table. `columnName7` identifies the column that begins the time period. `columnName8` identifies the column that ends the time period. Table 108 on page 432 lists valid values for the BUSINESS_TIME parameter (valid for DB2 Version 12.1 and later).

**ADD WITHOUT_OVERLAPS**

This option adds a constraint that ensures unique time values for the columns that identify the beginning and ending of the BUSINESS_TIME period.
DROP WITHOUT OVERLAPS

This option removes the constraint that ensures unique time values for the BUSINESS_TIME period.

Figure 160: ALTER Table statement (page 2)

DROP PERIOD BUSINESS_TIME

ALTER PERIOD SYSTEM_TIME(columnName 9, columnName 10)

ADD PERIOD SYSTEM_TIME(columnName 11, columnName 12)

DROP PERIOD SYSTEM_TIME

ARCHIVING_TABLE

VERSIONING_TABLE

MQTParameters

alterTableOptions

DROP PERIOD BUSINESS_TIME

This option removes the BUSINESS_TIME period from the table.

ALTER PERIOD SYSTEM_TIME(columnName 9, columnName 10)

Modifies the columns that are used as the beginning and end of the SYSTEM_TIME period.

ADD PERIOD SYSTEM_TIME(columnName 11, columnName 12)

This option specifies to add a SYSTEM_TIME period to the table.

columnName 11 identifies the column that begins the time period.

columnName 12 identifies the column that ends the time period.
DROP PERIOD SYSTEM_TIME

This option removes the SYSTEM_TIME period from the table.

VERSIONING TABLE(tableOwner.tableName)

This option adds a history table. Table 109 on page 434 lists valid values for the APPEND parameter.

Table 109: VERSIONING TABLE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableOwner.tableName</td>
<td>Specifies the name of the associated history table for a base table</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Indicates that a history table is not associated with a base table</td>
</tr>
</tbody>
</table>

ARCHIVING TABLE(tableOwner.tableName)

This option associates an archive table to an archive-enabled table. Table 110 on page 434 lists valid values for the APPEND parameter.

Table 110: ARCHIVING TABLE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableOwner.tableName</td>
<td>Specifies the name of the associated archive table for an archive-enabled table</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Indicates that an archive table is not associated with an archive-enabled table</td>
</tr>
</tbody>
</table>

APPEND

This option specifies whether to use append processing for the table. With append processing, DB2 attempts to place table rows in accordance with the value in the rows’ cluster key columns when inserting or loading data. Table 111 on page 434 lists valid values for the APPEND parameter.

Table 111: APPEND parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to use append processing.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to use append processing.</td>
</tr>
</tbody>
</table>

MQTParameters

This option specifies the parameters for the MQT (see “MQTParameters” on page 448).
**alterTableOptions**

Provides additional options for tables (see “alterTableOptions” on page 449).

**alterColumnSpecification**

`alterColumnSpecification` includes the following parameters.
Figure 161: alterColumnSpecification

**alterColumnSpecification**

```
( 
  builtInDataType
  TYPESCHEMA dataTypeSchema 
  TYPOENAME dataTypeName
  FOR BIT DATA
    SBCS 
    MIXED
  FIELDPROC procedureName
  ( )
  , constant
  NOFIELDPROC
  NOFIELDPROCParMS
  INLINE LENGTH integer
  nullParameters
  IDENTITY NO YES
  ( 
    START decimal
    INCREMENT integer
    CACHE integer
    CYCLE NO YES
    ORDER NO YES
    MAXVALUE decimal
      <DEFLT>
    MINVALUE decimal
      <DEFLT>
  )
  ROWCHANGE NO YES
  MOVE BEFORE columnName
  AFTER
  NAME columnName
  HIDDEN NO YES
)```
*builtInDataType*

This option defines the column as the specified DB2 data type.
Figure 162: `builtInDataType`

```
builtInDataType

CHAR
  ( - integer - )
VARCHAR ( - integer - )
LONG VARCHAR
CLOB
  ( - integer )
    G
    M
    K
INTEGER
SMALLINT
BIGINT
FLOAT
  ( - integer - )
DECIMAL
    34 16
    ( - integer , - integer )
DECIMAL
GRAPHIC
  ( - integer - )
VARGRAPHIC ( - integer - )
LONG VARGRAPHIC
DBCLOB
  ( - integer )
    G
    M
    K
BINARY
  ( - integer - )
VARBINARY ( - integer - )
BLOB
  ( - integer )
    G
    M
    K
```
For information about valid data types, see the IBM documentation.

**TYPESCHEMA** `dataTypeSchema`

This option specifies the schema type of the column.

**TYPENAME** `dataTypeName`

This option specifies the distinct type of the column.

**FOR** `subtype` **DATA**

(persistent and global temporary tables) For character (CHAR), varying-length character (VARCHAR), and character large object (CLOB) columns, defines the character subtype. Table 112 on page 439 lists valid values for the `subtype`.

---

**Note**

If the parent table space or database is defined as Unicode, and the column is created with an implicit subtype, the FOR `subtype` DATA parameter is omitted from the column definition.

---

**Table 112: subtype values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIT</td>
<td>Defines the subtype as a binary character set.</td>
</tr>
<tr>
<td>SBCS</td>
<td>Defines the subtype as a single-byte character set.</td>
</tr>
<tr>
<td>MIXED</td>
<td>Defines the subtype as a mixed double-byte and single-byte character set.</td>
</tr>
</tbody>
</table>

**FIELDPROC** `procedureName(constant, ...)`

This option defines a field procedure exit routine for the defined column.
**NOFIELDPROC**

This option specifies to remove an existing field procedure exit routine.

**NOFIELDPROCPARMS**

This option indicates to keep the same field procedure exit routine, but not to specify a list of parameters.

**INLINE LENGTH integer**

This option specifies the length for an inline LOB column. A value of 0 indicates that the column is not a LOB column, or that the LOB column does not have a length.

**nullParameters**

Provides additional options for null parameters (see “nullParameters” on page 443).

**IDENTITY**

This option indicates whether the table includes an identity column. Each table can have one identity column. Table 113 on page 440 lists valid values for the IDENTITY parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that the table does not include an identity column.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that the table includes an identity column.</td>
</tr>
</tbody>
</table>

If the table includes an identity column, the parameters in Table 114 on page 440 describe the column.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| START decimal | Specifies the starting value for the identity column.  
The default value is the minimum value for a column in ascending sequence or the maximum value for a column in descending sequence. |
|_INCREMENT integer | Specifies the amount of change in the values for the identity column.  
The value can be any positive or negative integer, including 0 (zero). The default value is 1. |
| CACHE integer | Specifies the number of a set of values that are stored for the identity column.  
A value of 0 (zero) is the equivalent of NO CACHE. |
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CYCLE</td>
<td>Indicates whether values should be generated after the maximum or minimum value is reached:</td>
</tr>
<tr>
<td></td>
<td>■ NO (default) indicates that values should not be generated after the maximum or minimum value is reached</td>
</tr>
<tr>
<td></td>
<td>■ YES indicates that values should be generated after the maximum or minimum value is reached</td>
</tr>
<tr>
<td>ORDER</td>
<td>Indicates whether values must be generated in the order in which they are requested:</td>
</tr>
<tr>
<td></td>
<td>■ NO (default) indicates that values should not be generated in the order in which they are requested</td>
</tr>
<tr>
<td></td>
<td>■ YES indicates that values must be generated in the order in which they are requested</td>
</tr>
<tr>
<td>MAXVALUE</td>
<td>Specifies the maximum value for an identity column, the maximum value can be equal to or greater than the minimum value.</td>
</tr>
<tr>
<td></td>
<td>■ decimal specifies the value.</td>
</tr>
<tr>
<td></td>
<td>■ &lt;DEFLT&gt; specifies a default value for the maximum value.</td>
</tr>
<tr>
<td></td>
<td>If the column is in ascending sequence, &lt;DEFLT&gt; is the largest value allowed. If the column is in descending sequence, the default value is either the starting value or -1 if the starting value is not specified. &lt;DEFLT&gt; is the equivalent of NOMAXVALUE.</td>
</tr>
<tr>
<td>MINVALUE</td>
<td>Specifies the minimum value for an identity column, the minimum value can be equal to or less than the maximum value:</td>
</tr>
<tr>
<td></td>
<td>■ decimal specifies the value</td>
</tr>
<tr>
<td></td>
<td>■ &lt;DEFLT&gt; specifies a default value for the minimum value</td>
</tr>
<tr>
<td></td>
<td>If the column is in descending sequence, &lt;DEFLT&gt; is the smallest value allowed. If the column is in ascending sequence, the value is either the starting value or 1 if the starting value is not specified. &lt;DEFLT&gt; is the equivalent of NOMINVALUE.</td>
</tr>
</tbody>
</table>

**ROWCHANGE**

This option indicates whether to generate the time stamp of the most recent change to a row in a column. The data type must be TIMESTAMP, and the
null parameter must be NOT NULL. Table 115 on page 442 lists valid values for the ROWCHANGE parameter.

Table 115: ROWCHANGE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that a value should not be generated for the time stamp.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that a value should be generated for the time stamp.</td>
</tr>
</tbody>
</table>

**MOVE BEFORE columnName**

This option moves the column before `columnName` (which must be the current name of an existing column). For example, if the table X.TAB contains the columns COL1, COL2, and COL3, then the following CDL statement would result in a column order of COL2, COLX, and COL3.

```sql
ALTER TABLE X.TAB
ALTER COLUMN COL1 (NAME COLX)
ALTER COLUMN COL2 (MOVE BEFORE COL1)
```

**Note**

The following CDL statement would result in an error, since COLA does not exist in the table before the invocation of the CDL statement.

```sql
ALTER TABLE X.TAB
ADD COLUMN COLA BEFORE COL1 (col_spec...)
ALTER COLUMN COL2 (MOVE BEFORE COLA)
```

**MOVE AFTER columnName**

This option moves the column after `columnName` (which must be the current name of an existing column).

**NAME columnName**

This option specifies the new name of the column.

**HIDDEN**

This option indicates whether the column can be "hidden" or excluded from the results of a SELECT * SQL statement. Table 116 on page 442 lists valid values for the HIDDEN parameter.

Table 116: HIDDEN parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that the column is visible in the results of the statement.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that the column is hidden from the results of the statement.</td>
</tr>
</tbody>
</table>

You cannot specify HIDDEN YES for a ROWID column.
**nullParameters**

nullParameters uses the following parameters.

Figure 163: nullParameters

nullParameters

```
NOT NULL

WITH DEFAULT | DEFAULTVALUE

NULL

DEFAULT | DEFAULTVALUE

constant

USER

CURRENT SQLID
```

**NOT NULL**

This option specifies that the column cannot contain any null values. Table 117 on page 443 lists valid values for the NOT NULL parameter.

Table 117: NOT NULL parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WITH DEFAULT</td>
<td>Specifies that the column has a default value</td>
</tr>
<tr>
<td></td>
<td>If a default value is not specified, CHANGE MANAGER uses a value assigned</td>
</tr>
<tr>
<td></td>
<td>by the system.</td>
</tr>
<tr>
<td>DEFAULTVALUE</td>
<td>Specifies the default value of the column:</td>
</tr>
<tr>
<td></td>
<td>- constant specifies a numeric or non-numeric constant as the default</td>
</tr>
<tr>
<td></td>
<td>value</td>
</tr>
<tr>
<td></td>
<td>- USER specifies the value of the CURRENT USER special register</td>
</tr>
<tr>
<td></td>
<td>- CURRENT SQLID specifies the value of the CURRENT SQLID special register</td>
</tr>
</tbody>
</table>

Table 117 on page 443 lists valid values for the NOT NULL parameter.
GENERATED

For an identity column or a ROWID column, indicates when the value for the column is generated by DB2:

- ALWAYS indicates that the value for the column is generated by DB2
- DEFAULT indicates that the value for the column is generated by DB2 when a value is not specified

NULL

This option specifies that the column can contain null values. Any column added to a global temporary table must have a default value of null. Table 118 on page 444 lists valid values for the NULL parameter.

Table 118: NULL parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>Specifies that the column has a default value.</td>
</tr>
<tr>
<td></td>
<td>If a default value is not specified, CHANGE MANAGER uses a value assigned by the system.</td>
</tr>
<tr>
<td>DEFAULTVALUE</td>
<td>Specifies the default value of the column:</td>
</tr>
<tr>
<td></td>
<td>constant specifies a numeric or non-numeric constant as the default value</td>
</tr>
<tr>
<td></td>
<td>USER specifies the value of the CURRENT USER special register</td>
</tr>
<tr>
<td></td>
<td>CURRENT SQLID specifies the value of the CURRENT SQLID special register</td>
</tr>
</tbody>
</table>
**addColumnSpecification**

*addColumnSpecification* includes the following parameters.

**Figure 164: addColumnSpecification**

For a description of these parameters, see “*alterColumnSpecification*” on page 435.
**partitionParameters**

*partitionParameters* contains the following parameters.

**Figure 165: partitionParameters**

**NUMPARTS** number

This option specifies the new number of partitions for the table. NUMPARTS=0 specifies a nonpartitioned table.

**ALTER PART** *partitionNumber* 1

For a partitioned table, alters *partitionNumber* 1 according to the replacement key limit list for the table and the new number for the *partitionNumber* 2 partition of the table. To alter parameters for a partitioned table, the *partitionNumber* 1 value must not be zero. For example, the following example changes the value for the VALUES attribute for the first partition of a partitioned table to 4536.

```
ALTER TABLE ...
ALTER PART 1 VALUES (4536)
```

Table 119 on page 446 lists the valid values for the ALTER PART parameter.

**Table 119: ALTER PART parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUES (constant, ...)</td>
<td>Specifies the values for the <em>partitionNumber</em> partition.</td>
</tr>
</tbody>
</table>
ADD PART BEFORE `partitionNumber` VALUES (`constant, ...`)

For a partitioned table, adds a new partition according to the replacement key limit list for the table. The partition is added before the specified `partitionNumber`.

ADD PART AFTER `partitionNumber` VALUES (`constant, ...`)

For a partitioned table, adds a new partition according to the replacement key limit list for the table. The partition is added after the specified `partitionNumber`.

DROP PART `partitionNumber`

For a partitioned table, drops the specified partition.

KEYCOLUMNS (`columnName order, ...`)

This option specifies the replacement column list that defines the key for this table. Table 120 on page 447 lists the valid values for the KEYCOLUMNS parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASC</td>
<td>Places the columns in ascending order.</td>
</tr>
<tr>
<td>DESC</td>
<td>Places the columns in descending order.</td>
</tr>
</tbody>
</table>

TCPART

This option indicates whether the partitioned table uses table-controlled partitioning. Table 121 on page 447 lists the valid values for the TCPART parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that the table uses table-controlled partitioning.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that the table uses index-controlled partitioning.</td>
</tr>
</tbody>
</table>
**MQTParameters**

*MQTParameters* uses the following parameters for a materialized query table.

**Figure 166: MQTParameters**

MQTParameters

- **PATH (schemaList)**
  This option specifies the path for the materialized query table.

- **TEXT (MQTTText)**
  This option specifies the text for the materialized query table.

- **AS SELECT subselect**
  This option defines the SQL subselect statement that is used for creating the materialized query table. The AS SELECT clause is required.

- **MAINTAINED BY**
  This option specifies whether the system or the user maintains data in the materialized query table. *Table 122 on page 448* lists valid values for the MAINTAINED BY parameter.

**Table 122: MAINTAINED BY parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>Specifies that the system maintains the materialized query table.</td>
</tr>
<tr>
<td>USER</td>
<td>Specifies that the user maintains the materialized query table.</td>
</tr>
</tbody>
</table>
**QUERYOPT**

This option specifies whether the materialized query table can be used to optimize queries. Table 123 on page 449 lists valid values for the QUERYOPT parameter.

Table 123: QUERYOPT parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies that the table can be used to optimize queries (in other words, enables optimization).</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies that the table cannot be used to optimize queries (in other words, disables optimization).</td>
</tr>
</tbody>
</table>

---

**alterTableOptions**

*alterTableOptions* contains the following parameters.

Figure 167: *alterTableOptions*

```
alterTableOptions

EDITPROC  procedureName

NOEDITPROC

PRIMARYKEY (columnName)

NOPRIMARYKEY

COMMENT 'string'

NOCOMMENT

LABEL 'string'

NOLABEL

^ COLCOMMENT columnName 'string'

^ NOCOLCOMMENT columnName

^ COLLABEL columnName 'string'

^ NOCOLLABEL columnName

OBID number

CCSID ASCII

EBCDIC

UNICODE
```
EDITPROC `proceduresName`

This option defines a new edit procedure.

NOEDITPROC

This option removes an existing edit procedure.

PRIMARYKEY (`columnName`, ...)

Creates a new primary key for the table, consisting of the columns that are specified in the column name list. This primary key definition either replaces an existing primary key or creates one if one does not already exist.

NOPRIMARYKEY

This option removes a primary key definition from a table where one exists.

COMMENT 'string'

This option defines a new comment for the table.

NOCOMMENT

This option removes an existing comment.

LABEL 'string'

This option defines a new label for the table.

NOLABEL

This option removes an existing label.

COLCOMMENT `columnName` 'string'

This option defines a comment 'string' on `columnName`.

NOCOLCOMMENT `columnName`

This option removes an existing column comment from `columnName`.

COLLABEL `columnName` 'string'

This option defines a label 'string' on `columnName`.

NOCOLLABEL `columnName`

This option removes an existing column label from `columnName`.
**OBID number**

This option defines the object identifier for a table in a read-only shared database.

**CCSID**

*(global temporary tables)* This option specifies the new encoding scheme of the table. Table 124 on page 451 lists valid values for the CCSID parameter.

**Table 124: CCSID parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>Specifies ASCII encoding.</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>Specifies EBCDIC encoding.</td>
</tr>
<tr>
<td>UNICODE</td>
<td>Specifies Unicode encoding.</td>
</tr>
</tbody>
</table>

**ALTER TABLESPACE statement**

The ALTER TABLESPACE statement specifies the changes that are necessary to make a remote table space identical to a local table space.
**Figure 168: ALTER TABLESPACE statement**

**ALTER TABLESPACE**

![Diagram of ALTER TABLESPACE statement]

- **DATABASE** `databaseName_2`
- **NAME** `tableSpaceName_2`
- **OWNER** `authorizationID`
- **CONVERT_TO_PARTITION_BY_GROWTH**
- **NUMPARTS** `number`
- **ALTER PART** `partitionNumber` `tableSpacePartitionParameters`
- **ADD PART** `BEFORE` `AFTER` `partitionNumber` `tableSpacePartitionParameters`
- **DROP PART** `partitionNumber`
- **LOCKPART** `YES` `NO`
- **ROTATE** `number`
- **LARGE** `YES` `NO`
- **MEMBER CLUSTER** `YES` `NO`
- **DSSIZE** `integer` `G`

**tableSpaceOptions**

**ALTER TABLESPACE `databaseName_1` .`tableSpaceName_1`**

This option specifies the fully qualified name of the table space to be changed.

**DATABASE `databaseName_2`**

This option specifies the new database for this table space.

**NAME `tableSpaceName_2`**

This option specifies the new name for this table space.

**OWNER `authorizationID`**

This option specifies the new owner of the table space.
CONVERT_TO_PARTITION_BY_GROWTH

This option indicates that the product is converting a table space to a partition-by-growth table space.

Numparts number

This option specifies the new number of partitions for the table space. Numparts 0 specifies a nonpartitioned table space.

ALTER PART partitionNumber tableSpacePartitionParameters

For a partitioned table space (Numparts > 1), modifies part partitionNumber according to the parameters in tableSpacePartitionParameters (see “tableSpacePartitionParameters” on page 455).

Note

For a nonpartitioned table space, partitionNumber must be 0.

ADD PART BEFORE partitionNumber tableSpacePartitionParameters

For a partitioned table, adds a new partition according to tableSpacePartitionParameters (see “tableSpacePartitionParameters” on page 455). The partition is added before the specified partitionNumber.

ADD PART AFTER partitionNumber tableSpacePartitionParameters

For a partitioned table, adds a new partition according to tableSpacePartitionParameters (see “tableSpacePartitionParameters” on page 455). The partition is added after the specified partitionNumber.

DROP PART partitionNumber

For a partitioned table space, drops the specified partition.

LOCKPART

This option indicates whether selective partition locking is used when locking a partitioned table space. Table 125 on page 453 lists valid values for the LOCKPART parameter.

Table 125: LOCKPART parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that selective partition locking is used.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that selective partition locking is not used.</td>
</tr>
</tbody>
</table>
**ROTA**TE number

For a partitioned table space, this option specifies the number of logical partitions that should be rotated from the first to the last.

---

**Note**

ROTA**TE is used in the ALTER TABLESPACE statement only when you convert an alter-type work ID to CDL.

---

**LARGE**

This option indicates whether the table space is a LARGE table space. Table 126 on page 454 lists valid values for the LARGE parameter.

**Table 126: LARGE parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>For a partitioned table space, indicates that each partition has a maximum partition size of 4 GB.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that the table space is not a LARGE table space.</td>
</tr>
</tbody>
</table>

**MEMBER CLUSTER**

This option indicates whether inserted data is clustered by the clustering index. Table 127 on page 454 lists valid values for the MEMBER CLUSTER parameter.

**Table 127: MEMBER CLUSTER parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that data is clustered based on the availability of space map pages.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that data is not clustered by the clustering index.</td>
</tr>
</tbody>
</table>

**DSSIZE integer G**

This option specifies the maximum size for each table space partition in gigabytes.
**tableSpacePartitionParameters**

`tableSpacePartitionParameters` uses the following parameters.

**Figure 169: tableSpacePartitionParameters**

![Diagram of tableSpacePartitionParameters parameters](image)

**VCAT catalogName**

This option specifies the volume catalog to be used for the table space.

**STOGROUP**

This option specifies the new default storage group for the table space. Table 128 on page 455 lists valid values for the STOGROUP parameter.

**Table 128: STOGROUP parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>storageGroupName</code></td>
<td>Specifies the name of the storage group.</td>
</tr>
<tr>
<td><code>&lt;DEFLT&gt;</code></td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;DEFLT&gt;</code> is a reserved keyword.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PRIQTY number</td>
<td>Specifies the primary space that is allocated for the DB2-defined data set.</td>
</tr>
<tr>
<td>SECQTY number</td>
<td>Specifies the secondary space that is allocated for the DB2-defined data set.</td>
</tr>
<tr>
<td>ERASE</td>
<td>Indicates whether the data sets can be erased when they are deleted by a statement that drops the table space.</td>
</tr>
<tr>
<td></td>
<td>■ NO (default value)</td>
</tr>
<tr>
<td></td>
<td>Specifies not to erase the data sets.</td>
</tr>
<tr>
<td></td>
<td>■ YES</td>
</tr>
<tr>
<td></td>
<td>Specifies to erase the data sets.</td>
</tr>
</tbody>
</table>

**FREEPAGE number**

This option specifies that a free page is added after every number of pages upon table space creation or reorganization.

**PCTFREE number**

This option specifies the percentage of free space left on each page of the table space upon creation or reorganization.

**TRACKMOD**

This option indicates whether modified pages are tracked in the space map pages of the table space. Table 129 on page 456 lists valid values for the TRACKMOD parameter.

**Table 129: TRACKMOD parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to track modified pages. TRACKMOD YES is the default.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies not to track modified pages</td>
</tr>
</tbody>
</table>

**COMPRESS**

This option indicates whether rows in the table space or partition are compressed. Table 130 on page 457 lists valid values for the COMPRESS parameter.
Table 130: COMPRESS parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to compress data. COMPRESS NO is the default.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to compress data</td>
</tr>
</tbody>
</table>

**GBPCACHE**

This option specifies which pages are cached to the group buffer pool for this partition. Table 131 on page 457 lists valid values for the GBPCACHE parameter.

Table 131: GBPCACHE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANGED</td>
<td>Specifies to cache updated pages to the buffer pool. GBPCACHE CHANGED is the default.</td>
</tr>
<tr>
<td>ALL</td>
<td>Specifies to cache all pages as they are read.</td>
</tr>
<tr>
<td>NONE</td>
<td>Specifies not to cache pages to the buffer pool.</td>
</tr>
</tbody>
</table>
**tableSpaceOptions**

*tableSpaceOptions* includes the following parameters.

**Figure 170: tableSpaceOptions**

![Diagram](image)

**BUFFERPOOL**

This option specifies the new default buffer pool for the database for the creation of the table space. If the BUFFERPOOL keyword is not specified, the DB2 default value is used. Table 132 on page 459 lists valid values for the BUFFERPOOL parameter.
Table 132: BUFFERPOOL parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufferPoolName</td>
<td>Specifies the name of the buffer pool.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

**LOCKSIZE**

This option specifies the size of the locks for the table space. Table 133 on page 459 lists valid values for the LOCKSIZE parameter.

Table 133: LOCKSIZE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANY</td>
<td>Specifies to use any lock size.</td>
</tr>
<tr>
<td>PAGE</td>
<td>Specifies to use page locks.</td>
</tr>
<tr>
<td>TABLE</td>
<td>Specifies to use table locks.</td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>Specifies to use table space locks.</td>
</tr>
<tr>
<td>ROW</td>
<td>Specifies to use row locks.</td>
</tr>
<tr>
<td>LOB</td>
<td>For LOB table spaces, specifies to use LOB locks.</td>
</tr>
</tbody>
</table>

**LOCKMAX**

This option specifies the maximum number of locks for the table space. Table 134 on page 459 lists valid values for the LOCKMAX parameter.

Table 134: LOCKMAX parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>integer</td>
<td>Specifies the number of page, row, or LOB locks.</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>Specifies the value used by DB2 for the number of locks.</td>
</tr>
</tbody>
</table>

**CLOSE**

This option indicates whether the index for the table space can be closed when the index is not being used. Table 135 on page 460 lists valid values for the CLOSE parameter.
Table 135: CLOSE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies that the index can be closed. CLOSE YES is the default.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies that the index cannot be closed.</td>
</tr>
</tbody>
</table>

MAXPARTITIONS integer

This option specifies the maximum number of partitions in a partition-by-growth table space. The valid range of values is 1 through 4096.

CCSID

This option specifies the encoding scheme of the table space. Table 136 on page 460 lists valid values for the CCSID parameter.

Table 136: CCSID parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>Specifies ASCII encoding.</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>Specifies EBCDIC encoding.</td>
</tr>
<tr>
<td>UNICODE</td>
<td>Specifies Unicode encoding.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. The default encoding scheme is determined in the installation options module with the CCSID keyword.</td>
</tr>
</tbody>
</table>

MAXROWS integer

This option specifies the maximum number of rows per page for the table space. The valid range of values is 1 through 255. This parameter does not apply to LOB table spaces.

LOG

This option indicates whether changes to data in a table space are written to a log. Table 137 on page 460 lists valid values for the LOG parameter.

Table 137: LOG parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to write changes to a log. LOG YES is the default.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies not to write changes to a log.</td>
</tr>
</tbody>
</table>
DEFINE

This option indicates when the data sets for the table space are created. Table 138 on page 461 lists valid values for the DEFINE parameter.

Table 138: DEFINE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that DB2 creates the data sets when it creates the table space. DEFINE YES is the default.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that DB2 creates the data sets when a row is inserted into a table.</td>
</tr>
</tbody>
</table>

SEGSIZE number

This option specifies the number of pages for each segment of a segmented table space. SEGSIZE 0 specifies a unsegmented table space.

**Note**

DB2 requires SEGSIZE to be between 4 and 64 pages and divisible by 4.

ALTER TRIGGER statement

The ALTER TRIGGER statement defines changes to be made to a DB2 trigger.
Figure 171: ALTER TRIGGER statement

**ALTER TRIGGER**

```
ALTER TRIGGER

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCHEMA</td>
<td>This option specifies the name of the trigger to be changed.</td>
</tr>
<tr>
<td>NAME</td>
<td>This option specifies the new schema of the trigger.</td>
</tr>
<tr>
<td>OWNER</td>
<td>This option specifies the name of the trigger.</td>
</tr>
<tr>
<td>TABLE</td>
<td></td>
</tr>
<tr>
<td>ACTIVATE</td>
<td></td>
</tr>
<tr>
<td>GRANULARITY</td>
<td></td>
</tr>
<tr>
<td>ONEVENT</td>
<td></td>
</tr>
<tr>
<td>NOREFOLDTB</td>
<td></td>
</tr>
<tr>
<td>REFNEWTB</td>
<td></td>
</tr>
<tr>
<td>NOREFOLDTB</td>
<td></td>
</tr>
<tr>
<td>REFNEWTB</td>
<td></td>
</tr>
<tr>
<td>PATH</td>
<td></td>
</tr>
<tr>
<td>COMMENT</td>
<td></td>
</tr>
<tr>
<td>NOCOMMENT</td>
<td></td>
</tr>
<tr>
<td>SECURED</td>
<td></td>
</tr>
<tr>
<td>NOT SECURED</td>
<td></td>
</tr>
<tr>
<td>TRIGGERTEXT</td>
<td></td>
</tr>
</tbody>
</table>
```

**ALTER TRIGGER** `schema_1.triggerName_1`

This option specifies the name of the trigger to be changed.

**SCHEMA** `schema_2`

This option specifies the new schema of the trigger.

**NAME** `triggerName_2`

This option specifies the name of the trigger.
OWNER `triggerOwner`

This option specifies the owner of the trigger.

TABLE `tableOwner.tableName`

This option specifies the triggering base table of the associated trigger.

ACTIVATE

This option specifies whether the trigger is activated before or after the triggering event. Table 139 on page 463 lists valid values for the ACTIVATE parameter.

Table 139: ACTIVATE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BEFORE</td>
<td>Specifies to activate the trigger before the event.</td>
</tr>
<tr>
<td>AFTER</td>
<td>Specifies to activate the trigger after the event.</td>
</tr>
</tbody>
</table>

GRANULARITY

This option specifies whether DB2 executes the triggered action for each row or for each statement. Table 140 on page 463 lists valid values for the GRANULARITY parameter.

Table 140: GRANULARITY parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROW</td>
<td>Specifies to execute the action for each row.</td>
</tr>
<tr>
<td>STATEMENT</td>
<td>Specifies to execute the action for each statement.</td>
</tr>
</tbody>
</table>

ONEVENT

This option specifies when DB2 executes the trigger. Table 141 on page 463 lists valid values for the ONEVENT parameter.

Table 141: ONEVENT parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSERT</td>
<td>Specifies to have DB2 execute the trigger when an insertion occurs.</td>
</tr>
<tr>
<td>DELETE</td>
<td>Specifies to have DB2 execute the trigger when a deletion occurs.</td>
</tr>
<tr>
<td>UPDATE ALLCOLUMNS</td>
<td>Specifies to have DB2 execute the trigger when an update occurs, and to update all of the columns in the table.</td>
</tr>
</tbody>
</table>
### Value | Description
--- | ---
UPDATE COLUMNS columnNameList | Specifies to have DB2 execute the trigger when an update occurs, and to update a list of columns in columnNameList.

**REFOLDCORR** correlationName

This option specifies the OLD AS correlation name.

**NOREFOLDCORR**

This option removes the OLD AS correlation name.

**REFNEWCORR** correlationName

This option specifies the NEW AS correlation name.

**NOREFNEWCORR**

This option removes the NEW AS correlation name.

**REFOLDTB** identifier

This option specifies the OLD _TABLE AS identifier.

**NOREFOLDTB**

This option removes the OLD _TABLE AS identifier.

**REFNEWTB** identifier

This option specifies the NEW_TABLE AS identifier.

**NOREFNEWTB**

This option removes the NEW_TABLE AS identifier.

**PATH** (schemaList)

This option specifies the path for the trigger.

**COMMENT 'string'**

This option specifies the comment for the trigger.

**NOCOMMENT**

This option removes the comment for the trigger.

**SECURED**

This option specifies that the trigger is secured.
NOT SECURED

This option specifies that the trigger is not secured

TRIGGERTEXT (triggerText)

This option specifies the triggered action for the trigger.

ALTER VIEW statement

The ALTER VIEW statement defines changes to be made to an existing view definition.
Figure 172: ALTER VIEW statement

ALTER VIEW

This option specifies the fully qualified name of the view to be modified.

QUALIFIED

This option indicates whether the view was created with a qualified name. Table 142 on page 467 lists valid values for the QUALIFIED parameter.
Table 142: QUALIFIED parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that the view was created with a qualified name.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that the view was not created with a qualified name.</td>
</tr>
</tbody>
</table>

**OWNER viewOwner 2**

This option specifies the new owner of the view.

**NAME viewName 2**

This option specifies the new name of the view.

**CHECK OPTION**

This option specifies the new check option for the view. Views that reference global temporary tables cannot specify CHECK OPTION. Table 143 on page 467 lists valid values for the CHECK OPTION parameter.

Table 143: CHECK OPTION parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCAL</td>
<td>Specifies that all search conditions are checked conditionally.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies that all search conditions are checked, regardless of the check options specified. CHECK OPTION YES creates a cascaded view.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies not to use the definition of the view to check operations that use the view.</td>
</tr>
</tbody>
</table>

**COMMENT 'string'**

This option defines a comment for the view.

**NOCOMMENT**

This option removes an existing comment.

**LABEL 'string'**

This option defines a label for the view.

**NOLABEL**

This option removes an existing label.
COLCOMMENT columnName 'string'

This option defines a comment 'string' on columnName.

NOCOLCOMMENT columnName

This option removes an existing column comment from columnName.

COLLABEL columnName 'string'

This option defines a label 'string' on columnName.

NOCOLLABEL columnName

This option removes an existing column label from columnName.

COLUMNS (columnName, ...)

This option specifies the names of the columns to be included in the view.

Note
The COLUMNS parameter is used when the view does not contain an explicit column list. The ALTER COLUMN, ADD COLUMN, and DROP COLUMN parameters are used when the view contains an explicit column list.

NOCOLUMNS

This option removes the column list from the view definition. Columns appear in the order that is specified in the SELECT clause.

ALTER COLUMN columnName 1 (NAME columnName 2)

This option specifies a new name for a column.

Note
The ALTER COLUMN parameter is used when the view contains an explicit column list. The COLUMNS statement is used when the view does not contain an explicit column list.

ALTER COLUMN columnName 1 (MOVE BEFORE columnName 3)

Moves a column before another column.

Note
The ALTER COLUMN parameter is used when the view contains an explicit column list. The COLUMNS statement is used when the view does not contain an explicit column list.
ALTER COLUMN column1 (MOVE AFTER column3)

Moves a column after another column.

---

*Note*

The ALTER COLUMN parameter is used when the view contains an explicit column list. The COLUMNS statement is used when the view does not contain an explicit column list.

---

ADD COLUMN column1 BEFORE column2

This option adds a new column before another column.

---

*Note*

The ADD COLUMN parameter is used when the view contains an explicit column list. The COLUMNS parameter is used when the view does not contain an explicit column list.

---

ADD COLUMN column1 AFTER column2

This option adds a new column after another column.

---

*Note*

The ADD COLUMN parameter is used when the view contains an explicit column list. The COLUMNS parameter is used when the view does not contain an explicit column list.

---

DROP COLUMN column

This option removes a column.

---

*Note*

The DROP COLUMN parameter is used when the view contains an explicit column list. The COLUMNS parameter is used when the view does not contain an explicit column list.

---

PATH (schemaList)

This option specifies the path for this view.

---

AS SELECT subselect

This option defines a new SQL subselect statement as the definition for the view. If specified, the new select statement must appear last in the ALTER VIEW statement, after any other clauses.
The CREATE ALIAS statement creates a new alias for a remote table.

**CREATE ALIAS statement**

**CREATE ALIAS**

```
CREATE ALIAS aliasOwner.aliasName
```

This option specifies the name of the alias to be created.

**SQLID SQLID**

```
This option specifies the current SQLID.
```

**TABLE**

```
TABLE locationID.tableOwner.tableName
  viewOwner.viewName

COMMENT 'string'
LABEL 'string'
```

This option specifies the fully qualified name or name of the table or view that the alias references. **Table 144 on page 470** lists valid values for the TABLE parameter.

**Table 144: TABLE parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>locationID.tableOwner.tableName</code></td>
<td>Specifies the fully qualified name of the table that the alias references.</td>
</tr>
<tr>
<td><code>tableOwner.tableName</code></td>
<td>Specifies the name of the table that the alias references.</td>
</tr>
<tr>
<td><code>locationID.viewOwner.viewName</code></td>
<td>Specifies the fully qualified name of the view that the alias references.</td>
</tr>
<tr>
<td><code>viewOwner.viewName</code></td>
<td>Specifies the name of a view that the alias references.</td>
</tr>
</tbody>
</table>
COMMENT ’string’

This option specifies a comment for the alias.

LABEL ’string’

This option specifies a label for the alias.

CREATE AUXILIARY TABLE statement

The CREATE AUXILIARY TABLE statement defines a new auxiliary table to the DB2 system.

Figure 174: CREATE AUXILIARY TABLE statement

CREATE AUXILIARY TABLE

CREATE AUXILIARY TABLE tableOwner 1 . tableName 1

DATABASE databaseName

TABLESPACE tableSpaceName

COMMENT ’string’

LABEL ’string’

REFERENCETB tableOwner 2 . tableName 2

REFERENCECOL columnName

PART number

CREATE AUXILIARY TABLE tableOwner 1 . tableName 1

This option specifies the fully qualified name of the auxiliary table to be created.

DATABASE databaseName

This option specifies the database for the auxiliary table.

TABLESPACE tableSpaceName

This option specifies the table space for the auxiliary table.

COMMENT ’string’

This option defines a comment for the auxiliary table.
LABEL 'string'

This option defines a label for the auxiliary table.

REFERENCETB tableOwner.tableName

This option specifies the base table.

REFERENCECOL columnName

This option specifies the large object (LOB) column in the base table.

PART number

This option indicates the partition number to which the auxiliary table points in the base table space. If the table space is not partitioned, then PART 0.

CREATE CHECK statement

The CREATE CHECK statement defines a check constraint for a table’s referential integrity checking.

Figure 175: CREATE CHECK statement

```sql
CREATE CHECK tableOwner.tableName

PATH (schemaList)

CHECK (constraintText)
```

This option specifies the name of the check constraint. CREATE CHECK tableOwner.tableName.checkConstraintName specifies the fully qualified name. If checkConstraintName is not specified, DB2 automatically generates a name.

PATH (schemaList)

This option specifies the path for the check constraint.

CHECK (constraintText)

This option specifies the check constraint text.
CREATE UNIQUE CONSTRAINT statement

The CREATE UNIQUE CONSTRAINT statement defines a unique constraint for a key in a table.

**Figure 176: CREATE UNIQUE CONSTRAINT statement**

```
CREATE UNIQUE CONSTRAINT
  tableOwner.tableName.
  uniqueConstraintName
  PRIMARYKEY YES NO
  KEYCOLUMNS ( columnName, ... )
  PERIOD BUSINESS_TIME NO YES
```

CREATE UNIQUE CONSTRAINT `tableOwner.tableName`.

This option specifies the name of the unique constraint.

**CREATE UNIQUE CONSTRAINT `tableOwner.tableName.uniqueConstraintName`**

This option specifies the fully qualified name.

**PRIMARYKEY**

This option indicates whether the unique constraint is a primary key or a unique key. *Table 145 on page 473* lists valid values for the PRIMARYKEY parameter.

**Table 145: PRIMARYKEY parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that the unique constraint is a primary key PRIMARYKEY YES is the default.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that the unique constraint is a unique key</td>
</tr>
</tbody>
</table>

**KEYCOLUMNS (`columnName`, ...)**

This option specifies the primary key columns that are included in the definition of the unique constraint.
PERIOD BUSINESS_TIME

This option indicates whether the BUSINESS_TIME values are unique. The starting and ending columns of BUSINESS_TIME are defined in the table on which the index is created. Table 145 on page 473 lists valid values for the PERIOD BUSINESS_TIME parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies that the period of time defined by the starting time and ending time for each row of the table do not overlap.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies that the period of time defined by the starting time and ending time for each row of the table can overlap.</td>
</tr>
</tbody>
</table>

CREATE DATABASE statement

The CREATE DATABASE statement creates a new database.

Figure 177: CREATE DATABASE statement
CREATE DATABASE databaseName

This option specifies the name of the new database.

OWNER authorizationID

This option specifies the owner of the database.

STOGROUP

This option specifies the default storage group for the database. If the STOGROUP parameter is not specified, the default DB2 SYSDEFLT storage group is used. Table 147 on page 475 lists valid values for the STOGROUP parameter.

Table 147: STOGROUP parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>storageGroupName</td>
<td>Specifies the name of the storage group.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

BUFFERPOOL

This option specifies the new default buffer pool for the database for the creation of the table space. If the BUFFERPOOL parameter is not specified, the DB2 default value is used. Table 148 on page 475 lists valid values for the BUFFERPOOL parameter.

Table 148: BUFFERPOOL parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufferPoolName</td>
<td>Specifies the name of the buffer pool.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

CCSID

This option specifies the encoding scheme of the database. Table 149 on page 475 lists valid values for the CCSID parameter.

Table 149: CCSID parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>Specifies ASCII encoding.</td>
</tr>
</tbody>
</table>
**INDEXBP**

This option specifies the default buffer pool for the database for the creation of the index. If the INDEXBP parameter is not specified, the DB2 default value is used. Table 150 on page 476 lists valid values for the INDEXBP.

**Table 150: INDEXBP parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufferPoolName</td>
<td>Specifies the name of the buffer pool.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

**CREATE FOREIGN KEY statement**

The CREATE FOREIGN KEY statement defines a foreign key for a table’s referential integrity checking.
Figure 178: CREATE FOREIGN KEY statement

CREATE FOREIGN KEY tableOwner.tableName

1. Specifies the name of the constraint for the foreign key. CREATE FOREIGN KEY tableOwner1.tableName1.constraintName specifies the fully qualified name. If constraintName is not specified, DB2 automatically generates a name.

REFERENCETB tableOwner2.tableName2

2. Specifies the name of the table to which the foreign key references.

KEYCOLUMNS (columnName, ...)

3. Specifies the columns that are included in the definition of the foreign key.

REFCOLUMNS (columnName, ...)

4. Specifies the names of the parent key columns.

Note

REFCOLUMNS should only be specified when the foreign key constraint name is not specified. To change the parent column list, alter the unique index that the foreign key uses.

ON DELETE

5. Specifies the new rule that determines the action to take when a row of the parent table for the foreign key is deleted. Table 151 on page 478 lists valid values for the ON DELETE parameter.
Table 151: ON DELETE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASCADE</td>
<td>Specifies a delete rule of CASCADE.</td>
</tr>
<tr>
<td>RESTRICT</td>
<td>Specifies a delete rule of RESTRICT if the value of the CURRENT RULES special register is DB2.</td>
</tr>
<tr>
<td>SET NULL</td>
<td>Specifies a delete rule of SET NULL if a column of the foreign key allows null values.</td>
</tr>
<tr>
<td>NO ACTION</td>
<td>Specifies a delete rule of NO ACTION if the value of the CURRENT RULES special register is SQL.</td>
</tr>
</tbody>
</table>

ENFORCED

Specifies whether DB2 enforces the referential constraint. Table 152 on page 478 lists valid values for the ENFORCED parameter.

Table 152: ENFORCED parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies that DB2 enforces the referential constraint.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies that DB2 does not enforce the referential constraint.</td>
</tr>
</tbody>
</table>

CREATE INDEX statement

The CREATE INDEX statement defines a new index for a table.
CREATE INDEX

CREATE INDEX \( indexOwner \_ indexName \_ \) SQLID SQLID

TABLE tableOwner \_ table\Name

KEYCOLUMNS (columnName ASC DESC RANDOM INCLUDE

KEYTEXT (expression

XML\DataType

PERIOD BUSINESS\_TIME NO YES

UNIQUE YES NO NOTNULL

AUX NO YES

COMMENT 'string'

NOCOMMENT

PART 0 indexPartitionParameters

NUMPARTS number ( PART number indexPartitionParameters )

indexOptions

CREATE INDEX indexOwner.indexName

This option specifies the fully qualified name of the new index.

SQLID SQLID

This option specifies the current SQLID.
TABLE `tableOwner$tableName`

This option specifies the table on which the index is being created. A global temporary table cannot be referenced in a CREATE INDEX statement.

KEYCOLUMNS (`columnName order, ...`)

This option specifies the column list that defines the key for this index. Table 153 on page 480 lists valid values for the `order` variable in the KEYCOLUMNS parameter.

Table 153: `order` values for the KEYCOLUMNS parameter

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASC</td>
<td>Places the key columns in ascending order. The key columns for indexes that are defined on an XML column and indexes on an expression can be placed only in ascending order.</td>
</tr>
<tr>
<td>DESC</td>
<td>Places the key columns in descending order.</td>
</tr>
<tr>
<td>RANDOM</td>
<td>Places the key columns in a random order.</td>
</tr>
<tr>
<td>INCLUDE</td>
<td>Specifies an additional column to be appended to the index key columns for a unique index.</td>
</tr>
</tbody>
</table>

KEYTEXT (`expression`)

This option specifies the parts of the XML documents in an XML column that are indexed (XMLPATTERN).

XMLdataType

For an XML index, this option specifies the SQL data type. The XMLdataType can be one of the following types:

- VARCHAR(`integer`)
- DECFLOAT
- DATE
- TIMESTAMP(12)

PERIOD BUSINESS_TIME

This option indicates whether the BUSINESS_TIME values are unique. The starting and ending columns of BUSINESS_TIME are defined in the table on which the index is created. Table 154 on page 481 lists valid values for the PERIOD BUSINESS_TIME parameter.
Table 154: PERIOD BUSINESS_TIME parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies that the period of time defined by the starting time and ending time for each row of the table do not overlap.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies that the period of time defined by the starting time and ending time for each row of the table can overlap.</td>
</tr>
</tbody>
</table>

**UNIQUE**

This option indicates whether index values must be unique. Table 155 on page 481 lists valid values for the UNIQUE parameter.

Table 155: UNIQUE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Defines the index as unique.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies to allow duplicates.</td>
</tr>
<tr>
<td>NOTNULL</td>
<td>Defines the index as unique and specifies that null values are not equal.</td>
</tr>
</tbody>
</table>

**AUX**

This option indicates whether the index is an auxiliary index for a large object (LOB) column in an auxiliary table. Table 156 on page 481 lists valid values for the AUX parameter.

Table 156: AUX parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that the index is not an auxiliary index.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that the index is an auxiliary index.</td>
</tr>
</tbody>
</table>

**COMMENT 'string'**

This option specifies the comment for the index.

**NOCOMMENT**

This option removes the comment for the index.

**PART 0 indexPartitionParameters**

This option specifies a nonpartitioned index according to indexPartitionParameters (see Figure 180 on page 482).
NUMPARTS number (PART number indexPartitionParameters)

Defines the number of partitions for the index. For a partitioned index, NUMPARTS defines the parameters for PART number according to indexPartitionParameters (see Figure 180 on page 482).

**indexOptions**

Provides additional options for indexes (see Figure 181 on page 484).

**indexPartitionParameters**

indexPartitionParameters includes the following parameters.

**Figure 180: indexPartitionParameters**

- **VCAT catalogName**
  
  This option specifies the volume catalog to be used for the index.

- **STOGROUP**

  This option specifies the new default storage group for the index. Table 157 on page 483 lists valid values for the STOGROUP parameter.
Table 157: STOGROUP parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>storageGroupName</td>
<td>Specifies the name of the storage group.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

**PRIQTY number**

This option specifies the primary space that is allocated for the DB2-defined data set.

**SECQTY number**

This option specifies the secondary space that is allocated for the DB2-defined data set.

**ERASE**

This option indicates whether to erase the data sets when they are deleted by a statement that drops the index. Table 158 on page 483 lists valid values for the ERASE parameter.

Table 158: ERASE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to erase the data sets.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies not to erase the data sets.</td>
</tr>
</tbody>
</table>

**FREEPAGE number**

This option specifies that a free page is added after every number of pages upon index creation or reorganization.

**PCTFREE number**

This option specifies the percentage of free space left on each page of the index upon creation or reorganization.

**GBPCACHE**

This option specifies which index pages are cached to the group buffer pool. Table 159 on page 484 lists valid values for the GBPCACHE parameter.
Table 159: GBPCACHE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANGED</td>
<td>Specifies to cache updated index pages to the buffer pool.</td>
</tr>
<tr>
<td>ALL</td>
<td>Specifies to cache all index pages as they are read.</td>
</tr>
<tr>
<td>NONE</td>
<td>Specifies not to cache index pages to the buffer pool.</td>
</tr>
</tbody>
</table>

VALUES (constant, ...)

This option specifies the replacement key limit list for this part of a partitioned index.

**indexOptions**

*indexOptions* includes the following parameters.

Figure 181: *indexOptions*

*indexOptions*
CLUSTER

This option indicates whether the index is defined as a clustering index. Table 160 on page 485 lists valid values for the CLUSTER parameter.

Table 160: CLUSTER parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to use the index as the clustering index for the table.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies not to use the index as the clustering index for the table.</td>
</tr>
</tbody>
</table>

PADDED

This option indicates how columns that are defined as VARCHAR are stored in the index. Table 161 on page 485 lists valid values for the PADDED parameter.

Table 161: PADDED parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that the columns should not be padded to the length of the index.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that the columns should be padded.</td>
</tr>
</tbody>
</table>

COMPRESS

This option indicates whether index data is compressed. Table 162 on page 485 lists valid values for the COMPRESS parameter.

Table 162: COMPRESS parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to use index compression.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to use index compression.</td>
</tr>
</tbody>
</table>

DEFINE

This option specifies when the data sets for the index are created. Table 163 on page 485 lists valid values for the DEFINE parameter.

Table 163: DEFINE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that DB2 creates the data sets when it alters the index. DEFINE YES is the default.</td>
</tr>
</tbody>
</table>
Value | Description
--- | ---
NO | Indicates that DB2 creates the data sets when a row is inserted into a table

**PARTITIONED**

This option indicates whether the index is partitioned. Table 164 on page 486 lists valid values for the PARTITIONED parameter.

**Table 164: PARTITIONED parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that the index is partitioned.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that the index is not partitioned.</td>
</tr>
</tbody>
</table>

**BUFFERPOOL**

This option specifies the new default buffer pool for the index. Table 165 on page 486 lists valid values for the BUFFERPOOL parameter.

**Table 165: BUFFERPOOL parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufferPoolName</td>
<td>Specifies the name of the buffer pool.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

**CLOSE**

This option indicates whether the index can be closed when the index is not being used. Table 166 on page 486 lists valid values for the CLOSE parameter.

**Table 166: CLOSE parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that the index can be closed. CLOSE YES is the default.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that the index cannot be closed.</td>
</tr>
</tbody>
</table>

**PIECESIZE integer keyword**

This option specifies the maximum PIECESIZE for a nonpartitioned index. The default PIECESIZE is 2 G (2 GB) for indexes backed by non-LARGE table spaces and 4 G (4 GB) for indexes backed by LARGE table spaces. The
subsequent keyword (K, G, or M) indicates the value of the specific integer (Table 167 on page 487).

### Table 167: PIECESIZE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>K</td>
<td>Specifies the PIECESIZE in kilobytes.</td>
</tr>
<tr>
<td>M</td>
<td>Specifies the PIECESIZE in megabytes.</td>
</tr>
<tr>
<td>G</td>
<td>Specifies the PIECESIZE in gigabytes.</td>
</tr>
</tbody>
</table>

**COPY**

This option indicates whether to use a utility to create an image copy of the index. Table 168 on page 487 lists valid values for the COPY parameter.

### Table 168: COPY parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| NO    | Indicates not to use a utility to create an image copy.  
      | COPY NO is the default.                          |
| YES   | Indicates to use a utility to create an image copy. |

**CREATE PROCEDURE statement**

The CREATE PROCEDURE statement defines a stored procedure.
Figure 182: CREATE PROCEDURE statement (page 1)

CREATE PROCEDURE

CREATE PROCEDURE — procedureOwner - , - procedureSchema - , - procedureName —

VERSION — versionNumber

ORIGIN — EXTERNAL

PARAMETERS — (ordinality, parmUsage, parmName, dataType, subType, locator —)

ACTIVE — NO

ADD_VERSION — YES

APPLCOMPAT

APPLICATION_ENCODING_SCHEME — ASCII

ARCHIVE SENSITIVE — YES

ASUTIME n

BUSINESS_TIME SENSITIVE — YES

COLLID — <NONE>

COMMENT — remarks — <NONE>

COMMIT_ON_RETURN

CONCURRENT_ACCESS_RESOLUTION — <DEFLT>

createProcedure page 2
CREATE PROCEDURE `procedureOwner`.`procedureSchema`.`procedureName`

This option specifies the fully qualified name of the new procedure.

VERSION `versionNumber`

This option specifies the version of the procedure.

ORIGIN

This option specifies the origin of the procedure. Table 169 on page 489 lists valid values for the ORIGIN keyword.

Table 169: ORIGIN keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXTERNAL</td>
<td>Specifies an external stored procedure.</td>
</tr>
<tr>
<td>NATIVE</td>
<td>Specifies a native SQL stored procedure.</td>
</tr>
</tbody>
</table>

PARAMETERS (ordinality, parmUsage, parmName, dataType, subType, locator)

This option specifies the parameters for a procedure. Table 170 on page 489 lists valid values for the procedure parameters.

Table 170: Procedure parameters values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ordinality</td>
<td>Specifies the ordinal number of the parameter.</td>
</tr>
<tr>
<td>parmUsage</td>
<td>Specifies whether the parameter is used as input, output, or both.</td>
</tr>
<tr>
<td>parmName</td>
<td>Specifies the name of the parameter.</td>
</tr>
<tr>
<td>dataType</td>
<td>Specifies the data type of the parameter.</td>
</tr>
<tr>
<td>subType</td>
<td>Specifies the subtype of the distinct data type.</td>
</tr>
<tr>
<td>locator</td>
<td>Indicates whether a locator to a value or the value itself is returned.</td>
</tr>
</tbody>
</table>

ACTIVE

This option indicates whether to activate the version of the procedure. Table 171 on page 490 lists valid values for the ACTIVE keyword.
Table 171: ACTIVE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that the version should not be recognized as the active version.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to activate the version.</td>
</tr>
</tbody>
</table>

ADD_VERSION YES

This option specifies to create a new version of the procedure when you LIKE a native SQL stored procedure.

APPLCOMPAT

This option indicates the package compatibility level for SQL.

Table 172 on page 490 lists valid values for the APPLCOMPAT keyword.

Table 172: APPLCOMPAT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>V10R1</td>
<td>Specifies that the SQL statements in the package are compatible with DB2 Version 10 Release 1.</td>
</tr>
<tr>
<td>V11R1</td>
<td>Specifies that the SQL statements in the package are compatible with DB2 Version 11 Release 1.</td>
</tr>
<tr>
<td>V12R1Mnnn</td>
<td>Specifies that the SQL statements in the package are compatible with DB2 Version 12 Release 1 Modification nnn.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

APPLICATION_ENCODING_SCHEME

This option specifies the encoding scheme for the procedure. Table 173 on page 490 lists valid values for the APPLICATION_ENCODING_SCHEME keyword.

Table 173: APPLICATION_ENCODING_SCHEME keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>Specifies ASCII encoding.</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>Specifies EBCDIC encoding.</td>
</tr>
<tr>
<td>UNICODE</td>
<td>Specifies Unicode encoding.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>
ARCHIVE SENSITIVE

This option specifies whether references to archive-enabled tables in SQL statements are affected by the value of the SYSIBMADM.GET_ARCHIVE global variable. Table 174 on page 491 lists valid values for the ARCHIVE SENSITIVE keyword.

Table 174: ARCHIVE SENSITIVE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that references are not affected.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies that references are affected.</td>
</tr>
</tbody>
</table>

ASUTIME \( n \)

This option specifies the number of CPU service units for the procedure. A value of 0 indicates that no limit exists on the number of units.

BUSINESS_TIME SENSITIVE

This option specifies whether references to application-period temporal tables in both static and dynamic SQL statements are affected by the value of the CURRENT TEMPORAL BUSINESS_TIME special register. Table 175 on page 491 lists valid values for the BUSINESS_TIME SENSITIVE keyword.

Table 175: BUSINESS_TIME SENSITIVE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that references are not affected.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies that references are affected.</td>
</tr>
</tbody>
</table>

COLLID

This option specifies the name of the package collection for the procedure. (The name of this collection is stored in the SYSIBM.SYSROUTINES table.) Table 176 on page 491 lists valid values for the COLLID keyword.

Table 176: COLLID keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;NONE&gt;</td>
<td>If a blank value is specified for the collection, specifies NO COLLID.</td>
</tr>
<tr>
<td>collectionID</td>
<td>Specifies the name of the package collection for the procedure.</td>
</tr>
</tbody>
</table>
COMMENT

This option specifies the comments for a procedure. Table 177 on page 492 lists valid values for the COMMENT keyword.

Table 177: COMMENT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>remarks</td>
<td>Specifies a comment for the procedure.</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies no comments for the procedure.</td>
</tr>
</tbody>
</table>

COMMIT_ON_RETURN NO

This option specifies that DB2 should continue the unit of work

COMMIT_ON_RETURN YES

This option specifies that DB2 should commit the unit of work immediately.

AUTONOMOUS

For native SQL procedures, this option specifies that DB2 should execute the SQL procedure in a unit of work that is independent from the calling application.

CONCURRENT_ACCESS_RESOLUTION

This option specifies the concurrent access resolution option to use for statements in a package. Table 178 on page 492 lists valid values for the CONCURRENT_ACCESS_RESOLUTION keyword.

Table 178: CONCURRENT_ACCESS_RESOLUTION keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
<tr>
<td>WAIT</td>
<td>Specifies the WAITFOROUTCOME option.</td>
</tr>
<tr>
<td>USE</td>
<td>Specifies the USECURRENTLYCOMMITTED option.</td>
</tr>
</tbody>
</table>
Figure 183: CREATE PROCEDURE statement (page 2)

createProcedure (page 2)

- CURRENT_DATA: NO, YES
- DATE_FORMAT: <DEFLT>, EUR, ISO, JIS, LOCAL, USA
- DBINFO: NO, YES
- DEBUG_MODE: ALLOW, DISALLOW, DISABLE, <DEFLT>
- DECIMAL: (x, y), <DEFLT>
- DEFER PREPARE: YES, NO, <DEFLT>
- DEGREE: 1, ANY
- DETERMINISTIC: NO, YES
- DYNAMICRULES: RUN, BIND, DEFINEBIND, DEFINERUN, INVOKEBIND, INVOKERUN
- EXPLAIN: NO, YES
- EXTERNAL_NAME: externalName, JAVAExternalName
- IMMEDIATE_WRITE: NO
CURRENT_DATA

This option indicates whether to require data currency for cursors when the isolation level of cursor stability is in effect. Table 179 on page 494 lists valid values for the CURRENT_DATA keyword.

Table 179: CURRENT_DATA keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies to not require data currency.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to require data currency.</td>
</tr>
</tbody>
</table>

DATE_FORMAT

This option specifies the format for the date for the procedure. Table 180 on page 494 lists valid values for the DATE_FORMAT keyword.

Table 180: DATE_FORMAT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
<tr>
<td>EUR</td>
<td>Specifies the IBM European standard (dd.mm.yyyy)</td>
</tr>
<tr>
<td>ISO</td>
<td>Specifies the International Standards Organization format (yyyy-mm-dd).</td>
</tr>
<tr>
<td>JIS</td>
<td>Specifies the Japanese industrial standard Christian era format (yyyy-mm-dd).</td>
</tr>
<tr>
<td>LOCAL</td>
<td>Specifies the installation-defined format.</td>
</tr>
<tr>
<td>USA</td>
<td>Specifies the IBM USA standard (mm/dd/yyyy).</td>
</tr>
</tbody>
</table>

DBINFO

This option indicates whether to pass the DBINFO structure to the stored procedure. Table 181 on page 494 lists valid values for the DBINFO keyword.

Table 181: DBINFO keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to pass the DBINFO structure to the stored procedure.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to pass the DBINFO structure to the stored procedure.</td>
</tr>
</tbody>
</table>
DEBUG_MODE

This option specifies whether the procedure is allowed for debugging. Table 182 on page 495 lists valid values for the DEBUG_MODE keyword.

Table 182: DEBUG_MODE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLOW</td>
<td>Specifies to allow debugging.</td>
</tr>
<tr>
<td>DISALLOW</td>
<td>Specifies not to allow debugging.</td>
</tr>
<tr>
<td>DISABLE</td>
<td>Specifies to disable debugging.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

DECIMAL

This option specifies the rules for a decimal operation (DEC31 or DEC15). Table 183 on page 495 lists valid values for the DECIMAL keyword.

Table 183: DECIMAL keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x, y)</td>
<td>Specifies the decimal value (x) and the scale (y). The decimal value is either 31 or 15. Values for scale are 1 through 9.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

DEFER_PREPARE

This option indicates whether to immediately prepare dynamic SQL statements or to defer preparation. Table 184 on page 495 lists valid values for the DEFER_PREPARE keyword.

Table 184: DEFER_PREPARE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to defer the preparation of dynamic SQL statements.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies to immediately prepare dynamic SQL statements.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value or to inherit the option from the plan.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>
DEGREE

This option specifies whether to attempt to run a query using parallel processing. Table 185 on page 496 lists valid values for the DEGREE keyword.

Table 185: DEGREE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Specifies to prohibit parallel processing.</td>
</tr>
<tr>
<td>ANY</td>
<td>Specifies to allow parallel processing.</td>
</tr>
</tbody>
</table>

DETERMINISTIC

This option indicates whether to use indeterminate or deterministic results for a procedure. Table 186 on page 496 lists valid values for the DETERMINISTIC keyword.

Table 186: DETERMINISTIC keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies to use indeterminate results.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to use deterministic results.</td>
</tr>
</tbody>
</table>

DYNAMICRULES

This option specifies the values for the dynamic SQL attributes. Table 187 on page 496 lists valid values for the DYNAMICRULES keyword.

Table 187: DYNAMICRULES keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUN</td>
<td>Specifies to process dynamic SQL statements using standard attribute values.</td>
</tr>
<tr>
<td>BIND</td>
<td>Specifies to process dynamic SQL statements using bind behavior.</td>
</tr>
<tr>
<td>DEFINEBIND</td>
<td>Specifies to process dynamic SQL statements using define behavior or bind behavior.</td>
</tr>
<tr>
<td>DEFINERUN</td>
<td>Specifies to process dynamic SQL statements using define behavior or run behavior.</td>
</tr>
<tr>
<td>INVOKEBIND</td>
<td>Specifies to process dynamic SQL statements using invoke behavior or bind behavior.</td>
</tr>
<tr>
<td>INVOKERUN</td>
<td>Specifies to process dynamic SQL statements using invoke behavior or run behavior.</td>
</tr>
</tbody>
</table>
EXPLAIN

This option indicates whether to add information about the package’s statements to the owner of the PLAN_TABLE table. Table 188 on page 497 lists valid values for the EXPLAIN keyword.

Table 188: EXPLAIN keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to add information.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to add information.</td>
</tr>
</tbody>
</table>

EXTERNAL_NAME externalName JAVAExternalName

This option specifies the module that DB2 should load to execute the procedure. For parameters for a JAVAExternalName, see “JAVAExternalName” on page 510.

IMMEDIATE_WRITE

This option indicates whether updates that are made to dependent page sets for group buffer pools should be immediately written. Table 189 on page 497 lists valid values for the IMMEDIATE_WRITE keyword.

Table 189: IMMEDIATE_WRITE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies normal write activity.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to immediately write updated pages.</td>
</tr>
</tbody>
</table>
CREATE PROCEDURE statement (page 3)

```
createProcedure (page 3)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISOLATION_LEVEL</td>
<td>CS, RR, RS, UR</td>
</tr>
<tr>
<td>KEEP_DYNAMIC</td>
<td>NO, YES</td>
</tr>
<tr>
<td>LANGUAGE</td>
<td>ASSEMBLE, C, COBOL, PLI, REXX, SQL, JAVA</td>
</tr>
<tr>
<td>MAXIMUM_FAILURES</td>
<td>-1, 0, n</td>
</tr>
<tr>
<td>OPPTHINT</td>
<td>'value', &lt;NONE&gt;</td>
</tr>
<tr>
<td>OWNER</td>
<td>authorizationID</td>
</tr>
<tr>
<td>PACKAGE_OWNER</td>
<td>owner, &lt;NONE&gt;</td>
</tr>
<tr>
<td>PACKAGE_PATH</td>
<td>&lt;NONE&gt;, 'value1,value2,value3'</td>
</tr>
<tr>
<td>PARAMETER_CCSID</td>
<td>ASCII, EBCDIC, UNICODE, &lt;DEFLT&gt;</td>
</tr>
<tr>
<td>PARAMETER_STYLE</td>
<td>SQL, GENERAL_NO_NULLS, GENERAL_WITH_NULLS, JAVA</td>
</tr>
<tr>
<td>PARAMETER_VARCHAR</td>
<td>&lt;DEFLT&gt;, NULLTERM, STRUCTURE</td>
</tr>
<tr>
<td>PROGRAM_TYPE</td>
<td>SUB, MAIN</td>
</tr>
</tbody>
</table>
```
**ISOLATION_LEVEL**

This option specifies the isolation level. Table 190 on page 499 lists valid values for the ISOLATION_LEVEL keyword.

**Table 190: ISOLATION_LEVEL keyword values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>Specifies cursor stability.</td>
</tr>
<tr>
<td>RR</td>
<td>Specifies repeatable read.</td>
</tr>
<tr>
<td>RS</td>
<td>Specifies read stability.</td>
</tr>
<tr>
<td>UR</td>
<td>Specifies uncommitted read.</td>
</tr>
</tbody>
</table>

**KEEP_DYNAMIC**

This option indicates whether dynamic SQL statements are purged at each commit point. Table 191 on page 499 lists valid values for the KEEP_DYNAMIC keyword.

**Table 191: KEEP_DYNAMIC keyword values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies to purge dynamic SQL statements.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to keep dynamic SQL statements.</td>
</tr>
</tbody>
</table>

**LANGUAGE**

This option specifies the implementation language of the procedure. Table 192 on page 499 lists valid values for the LANGUAGE keyword.

**Table 192: LANGUAGE keyword values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSEMBLE</td>
<td>Specifies assembler.</td>
</tr>
<tr>
<td>C</td>
<td>Specifies C.</td>
</tr>
<tr>
<td>COBOL</td>
<td>Specifies COBO.</td>
</tr>
<tr>
<td>PLI</td>
<td>Specifies PL/1.</td>
</tr>
<tr>
<td>REXX</td>
<td>Specifies REXX.</td>
</tr>
<tr>
<td>SQL</td>
<td>Specifies SQL.</td>
</tr>
<tr>
<td>JAVA</td>
<td>Specifies JAVA.</td>
</tr>
</tbody>
</table>
MAXIMUM_FAILURES

This option specifies the number of allowable failures for the procedure. Table 193 on page 500 lists valid values for the MAXIMUM_FAILURES keyword.

Table 193: MAXIMUM_FAILURES keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>Specifies to use the DB2 installation parameter (STORMXAB).</td>
</tr>
<tr>
<td>0</td>
<td>Specifies to never stop the procedure.</td>
</tr>
<tr>
<td>$n$</td>
<td>Specifies the number of failures. Values are 1 through 32767.</td>
</tr>
</tbody>
</table>

OPTHINT

This option specifies the value of the query optimization hint bind option. Table 194 on page 500 lists valid values for the OPTHINT keyword.

Table 194: OPTHINT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>'value'</td>
<td>Specifies a character string to be used by the optimizer when searching for input to DB2.</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies that DB2 should not use optimization hints for SQL statements.</td>
</tr>
</tbody>
</table>

OWNER authorizationID

This option specifies the owner of the procedure. The OWNER parameter is included in the CDL statement only if the OVERRIDE(SPOWNER) keyword is in the ALUIN input stream.

PACKAGE_OWNER

Specifies the owner of the package. Table 195 on page 500 lists valid values for the PACKAGE_OWNER keyword.

Table 195: PACKAGE_OWNER keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>Specifies the authorization ID of the owner of the package.</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies no owner for the package.</td>
</tr>
</tbody>
</table>
PACKAGE_PATH

This option specifies the package path to use when the procedure executes. Table 196 on page 501 lists valid values for the PACKAGE_PATH keyword.

Table 196: PACKAGE_PATH keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies the package path of the program that invoked the procedure.</td>
</tr>
<tr>
<td>'value1, value2, value3'</td>
<td>Specifies the package path..</td>
</tr>
</tbody>
</table>

PARAMETER_CCSID

This option specifies the encoding scheme for the parameters for the procedure. Table 197 on page 501 lists valid values for the PARAMETER_CCSID keyword.

Table 197: PARAMETER_CCSID keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>Specifies ASCII encoding.</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>Specifies EBCDIC encoding.</td>
</tr>
<tr>
<td>UNICODE</td>
<td>Specifies Unicode encoding.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

PARAMETERSTYLE

This option specifies the convention used to pass parameters to and return values from the procedure. Table 198 on page 501 lists valid values for the PARAMETER_STYLE keyword.

Table 198: PARAMETER_STYLE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL</td>
<td>Specifies to pass parameters according to the DB2SQL standard convention.</td>
</tr>
<tr>
<td>GENERAL_NO_NULLS</td>
<td>Specifies to pass parameters according to the GENERAL standard convention.</td>
</tr>
<tr>
<td>GENERAL_WITH_NULLS</td>
<td>Specifies to pass parameters according to the GENERAL WITH NULLS standard convention.</td>
</tr>
<tr>
<td>JAVA</td>
<td>Specifies to pass parameters according to the JAVA and SQLJ standard conventions.</td>
</tr>
</tbody>
</table>
PARAMETER_VARCHAR

This option specifies the representation of variable length string parameters. Table 199 on page 502 lists valid values for the PARAMETER_VARCHAR keyword.

Table 199: PARAMETER_VARCHAR keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td>NULLTERM</td>
<td>Specifies a null-terminated string form.</td>
</tr>
<tr>
<td>STRUCTURE</td>
<td>Specifies a VARCHAR structure form.</td>
</tr>
</tbody>
</table>

PROGRAM_TYPE

This option indicates whether the procedure runs as a main routine or a subroutine. Table 200 on page 502 lists valid values for the PROGRAM_TYPE keyword.

Table 200: PROGRAM_TYPE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUB</td>
<td>Specifies to run the procedure as a subroutine.</td>
</tr>
<tr>
<td>MAIN</td>
<td>Specifies to run the procedure as a main routine.</td>
</tr>
</tbody>
</table>

CDL statements
createProcedure (page 4)

- QUALIFIER  
  - qualifier  
    - <DEFLT>
- RELEASE_AT  
  - COMMIT
  - DEALLOCATE
- RESULT_SETS  
  - value
- REOPT  
  - NONE
  - ALWAYS
  - ONCE
- ROUNING  
  - DEC_ROUND_CEILING
  - DEC_ROUND_DOWN
  - DEC_ROUND_FLOOR
  - DEC_ROUND_HALF_DOWN
  - DEC_ROUND_HALF_EVEN
  - DEC_ROUND_HALF_UP
  - DEC_ROUND_UP
  - <DEFLT>
- RUN_OPTIONS  
  - runtimeOptions
    - <NONE>
- SECURITY  
  - DB2
  - USER
  - DEFINER
- SPECIAL_REGISTERS  
  - INHERIT
  - DEFAULT
- SQL_DATA_ACCESS  
  - MODIFIES
  - READS
  - CONTAINS
  - NONE
- SQL_PATH  
  - DEFAULT
  - USER
  - schemaName
  - schemaNameList
  - <NONE>

createProcedure page 5
QUALIFIER

This option specifies the implicit qualifier for the unqualified object names in the SQL statements. Table 201 on page 504 lists valid values for the QUALIFIER keyword.

Table 201: QUALIFIER keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>qualifier</td>
<td>Specifies the qualifier for the unqualified table, view, index, and alias names.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

RELEASE_AT

This option specifies the release resources when the package was last bound or rebound. Table 202 on page 504 lists valid values for the RELEASE_AT keyword.

Table 202: RELEASE_AT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMIT</td>
<td>Specifies the COMMIT value.</td>
</tr>
<tr>
<td>DEALLOCATE</td>
<td>Specifies the DEALLOCATE value.</td>
</tr>
</tbody>
</table>

RESULT_SETS value

This option specifies the number of query result sets that the procedure can return. Values are 0 through 32767.

REOPT

This option specifies whether to have DB2 determine an access path at run time. Table 203 on page 504 lists valid values for the REOPT keyword.

Table 203: REOPT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>Specifies to determine an access path at run time.</td>
</tr>
<tr>
<td>ALWAYS</td>
<td>Specifies to determine the access path at run time each time that the statement runs.</td>
</tr>
<tr>
<td>ONCE</td>
<td>Specifies to determine the access path only once.</td>
</tr>
</tbody>
</table>
ROUNDSING

This option specifies the rounding mode when the package is bound. Table 204 on page 505 lists valid values for the ROUNDSING keyword.

Table 204: ROUNDSING keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEC_ROUNDS_CEILING</td>
<td>Specifies to round toward infinity.</td>
</tr>
<tr>
<td>DEC_ROUNDS_DOWN</td>
<td>Specifies to round toward zero.</td>
</tr>
<tr>
<td>DEC_ROUNDS_FLOOR</td>
<td>Specifies to round toward negative infinity.</td>
</tr>
<tr>
<td>DEC_ROUNDS_HALF_DOWN</td>
<td>Specifies to round to the nearest digit. If the digit is equidistant, round down.</td>
</tr>
<tr>
<td>DEC_ROUNDS_HALF_EVEN</td>
<td>Specifies to round to the nearest digit. If the digit is equidistant, round to an even digit.</td>
</tr>
<tr>
<td>DEC_ROUNDS_HALF_UP</td>
<td>Specifies to round to the nearest digit. If the digit is equidistant, round up.</td>
</tr>
<tr>
<td>DEC_ROUNDS_UP</td>
<td>Specifies to round away from zero.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

RUN_OPTIONS

This option specifies the language environment run-time options for the procedure. Table 205 on page 505 lists valid values for the RUN_OPTIONS keyword.

Table 205: RUN_OPTIONS keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>runtimeOptions</td>
<td>Specifies the run-time options.</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies to use the default run-time options.</td>
</tr>
</tbody>
</table>

SECURITY

If the procedure accesses resources that are protected by an external security product, specifies the authorization ID. Table 206 on page 506 lists valid values for the SECURITY keyword.
Table 206: SECURITY keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB2</td>
<td>Specifies the authorization ID for the WLM procedure address space.</td>
</tr>
<tr>
<td>USER</td>
<td>Specifies the authorization ID of the user who invoked the procedure.</td>
</tr>
<tr>
<td>DEFINER</td>
<td>Specifies the authorization ID of the owner.</td>
</tr>
</tbody>
</table>

SPECIAL_REGISTERS

This option specifies how special registers are set. Table 207 on page 506 lists valid values for the SPECIAL_REGISTERS keyword.

Table 207: SPECIAL_REGISTERS keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INHERIT</td>
<td>Specifies that the values of special registers are inherited.</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>Specifies that the values of special registers are initialized to the default values.</td>
</tr>
</tbody>
</table>

SQL_DATA_ACCESS

This option specifies which SQL statements can be executed in the procedure. Table 208 on page 506 lists valid values for the SQL_DATA_ACCESS keyword.

Table 208: SQL_DATA_ACCESS keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODIFIES</td>
<td>Specifies that the procedure can execute only supported SQL statements.</td>
</tr>
<tr>
<td>READS</td>
<td>Specifies that the procedure can execute SQL statements with READS SQL DATA, CONTAINS SQL, or NO SQL.</td>
</tr>
<tr>
<td>CONTAINS</td>
<td>Specifies that the procedure can execute SQL statements with CONTAINS SQL or NO SQL.</td>
</tr>
<tr>
<td>NONE</td>
<td>Specifies that the procedure can execute SQL statements with NO SQL.</td>
</tr>
</tbody>
</table>
**SQL_PATH**

This option specifies the SQL path that DB2 uses to resolve unqualified types and names in the procedure. *Table 209 on page 508* lists valid values for the SQL_PATH keyword.
Table 209: SQL_PATH keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>Specifies &quot;SYSIBM&quot;, &quot;SYSFUN&quot;, &quot;SYSPROC&quot;, and the value of the QUALIFIER option.</td>
</tr>
<tr>
<td>USER</td>
<td>Specifies the value of the SESSION_USER or USER special register.</td>
</tr>
<tr>
<td>schemaName</td>
<td>Specifies the name of the schema.</td>
</tr>
<tr>
<td>schemaNameList</td>
<td>Specifies a list of schema names, separated by commas.</td>
</tr>
<tr>
<td>&lt;NONE&gt;</td>
<td>Specifies no SQL path</td>
</tr>
</tbody>
</table>

**STAY_RESIDENT**

This option indicates whether the procedure load module remains resident in memory when the procedure ends. Table 210 on page 508 lists valid values for the STAY_RESIDENT keyword.

Table 210: STAY_RESIDENT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies to delete the load module from memory.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to retain the load module in memory.</td>
</tr>
</tbody>
</table>

**SYSTEM_TIME SENSITIVE**

This option indicates whether references to system-period temporal tables in both static and dynamic SQL statements are affected by the value of the CURRENT TEMPORAL SYSTEM_TIME special register. Table 211 on page 508 lists valid values for the SYSTEM_TIME SENSITIVE keyword.

Table 211: SYSTEM_TIME SENSITIVE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that references are not affected.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies that references are affected.</td>
</tr>
</tbody>
</table>

**TIME_FORMAT**

This option specifies the time format for values. Table 212 on page 509 lists valid values for the TIME_FORMAT keyword.
Table 212: TIME_FORMAT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
<tr>
<td>EUR</td>
<td>Specifies the IBM European standard (hh:mm:ss).</td>
</tr>
<tr>
<td>ISO</td>
<td>Specifies the International Standards Organization format (hh:mm:ss).</td>
</tr>
<tr>
<td>JIS</td>
<td>Specifies the Japanese industrial standard Christian era format (hh:mm:ss).</td>
</tr>
<tr>
<td>LOCAL</td>
<td>Specifies the installation-defined format.</td>
</tr>
<tr>
<td>USA</td>
<td>Specifies the IBM USA standard (hh:mm AM or PM).</td>
</tr>
</tbody>
</table>

FOR_UPDATE_CLAUSE

This option specifies whether the FOR UPDATE clause is required or optional for a DECLARE CURSOR statement. Table 213 on page 509 lists valid values for the FOR_UPDATE_CLAUSE keyword.

Table 213: FOR_UPDATE_CLAUSE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQUIRED</td>
<td>Specifies that the clause is required.</td>
</tr>
<tr>
<td>OPTIONAL</td>
<td>Specifies that the clause is optional.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value. &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

VALIDATE

This option specifies whether to recheck errors. Table 214 on page 509 lists valid values for the VALIDATE keyword.

Table 214: VALIDATE keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUN</td>
<td>Specifies to issue a warning when the CREATE statement succeeds.</td>
</tr>
<tr>
<td>BIND</td>
<td>Specifies to issue an error when the CREATE statement fails.</td>
</tr>
</tbody>
</table>

WLM_ENVIRONMENT

This option specifies the workload manager (WLM) environment in which the procedure runs. Table 215 on page 510 lists valid values for the WLM_ENVIRONMENT keyword.
Table 215: WLM_ENVIRONMENT keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value.</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
<tr>
<td>WLMName</td>
<td>Specifies the name of the WLM environment.</td>
</tr>
</tbody>
</table>

WLM_ENV_FOR_NESTED

For nested procedures, this option indicates which address space DB2 uses. Table 216 on page 510 lists valid values for the WLM_ENV_FOR_NESTED keyword.

Table 216: WLM_ENV_FOR_NESTED keyword values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to use the address space specified in the WLM environment.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies to use an address space other than the one specified in the WLM environment.</td>
</tr>
</tbody>
</table>

TEXT storedProcedureBody

This option specifies the statements in the body of the SQL procedure.

JAVAExternalName

When the procedure is a JAVA procedure, JAVAExternalName uses the following parameters for an external name.

Figure 187: JAVAExternalName

JAVAExternalName

```
JAR_SCHEMA    --> schema
JAR_ID        --> ID
EXTERNAL_NAME --> name
JAVASIGNATURE --> signature
```
JAR_SCHEMA `schema`  
This option specifies the schema of the JAR file.

JAR_ID ID  
This option specifies the name of the JAR file.

EXTERNAL_NAME `name`  
This option specifies the `package.class.method` that DB2 should load to execute the JAVA procedure.

JAVA/SIGNATURE `signature`  
This option specifies the signature of the JAR file.

**CREATE SEQUENCE statement**

The CREATE SEQUENCE statement defines a sequence.
CREATE SEQUENCE schema.sequenceName

This option specifies the fully qualified name of the sequence.

TYPESCHEMA dataTypeSchema

This option specifies the schema for the sequence data type.

TYPENAME dataTypeName

This option specifies the name of the sequence data type.

OWNER ownerID

This option specifies the owner of the sequence.
**dataType**

This option specifies the data type of the sequence. The data type can be SMALLINT, INTEGER, BIGINT, DECIMAL, or a user-defined distinct type. The default is INTEGER.

**LENGTH numeric**

For sequences that are defined as a DECIMAL data type, this option specifies the length of the value.

**START numeric**

This option specifies the starting value for the sequence. The default value is the minimum value for an ascending sequence or the maximum value for a descending sequence.

**INCREMENT numeric**

This option specifies the amount of change in the values for the sequence. The value can be any positive or negative integer, including 0 (zero). The default value is 1.

**MINVALUE**

This option specifies the minimum value for a sequence. The minimum value can be equal to or less than the maximum value. Table 217 on page 513 lists valid values for the MINVALUE parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numeric</td>
<td>Specifies the minimum value.</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value for the minimum value.</td>
</tr>
<tr>
<td></td>
<td>If the sequence is descending, &lt;DEFLT&gt; is the smallest value allowed. If the sequence is ascending, the value is either the starting value or 1 if the starting value is not specified. &lt;DEFLT&gt; is the equivalent of NOMINVALUE.</td>
</tr>
</tbody>
</table>

**MAXVALUE**

This option specifies the maximum value for a sequence. The maximum value can be equal to or greater than the minimum value. Table 218 on page 513 lists valid values for the MAXVALUE parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numeric</td>
<td>Specifies the maximum value.</td>
</tr>
</tbody>
</table>
**Value** | **Description**
--- | ---
<DEFLT> | Specifies a default value for the maximum value. If the sequence is ascending, <DEFLT> is the largest value allowed. If the column is descending, the default value is either the starting value or -1 if the starting value is not specified. <DEFLT> is the equivalent of NOMAXVALUE.

**CYCLE**

This option indicates whether values should be generated after the maximum or minimum value is reached. Table 219 on page 514 lists valid values for the CYCLE parameter.

**Table 219: CYCLE parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that values should not be generated after the maximum or minimum value is reached. CYCLE NO is the default.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that values should be generated after the maximum or minimum value is reached.</td>
</tr>
</tbody>
</table>

**CACHE integer**

This option specifies the number of a set of values that are stored for the sequence. A value of 0 (zero) is the equivalent of NO CACHE.

**ORDER**

This option indicates whether values must be generated in the order in which they are requested. Table 220 on page 514 lists valid values for the ORDER parameter.

**Table 220: ORDER parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that values should not be generated in the order in which they are requested. ORDER NO is the default.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that values must be generated in the order in which they are requested.</td>
</tr>
</tbody>
</table>

**COMMENT 'string'**

This option specifies the comment for the sequence.
CREATE STOGROUP statement

The CREATE STOGROUP statement defines a new storage group to DB2.

**Figure 189: CREATE STOGROUP statement**

```
CREATE STOGROUP

CREATE STOGROUP  storageGroupName

VCAT  catalogName

VOLUMES  ( volumeSerialNumber, ... )

OWNER  authorizationID

DATACLAS  name
NODATACLAS

MGMTCLAS  name
NOMGMTCLAS

STORCLAS  name
NOSTORCLAS
```

**CREATE STOGROUP** `storageGroupName`

This option specifies the name of the storage group.

**VCAT** `catalogName`

This option specifies the volume catalog for the storage group.

**VOLUMES** `(volumeSerialNumber, ...)`

This option defines a list of volumes that are included in the storage group.

**OWNER** `authorizationID`

This option specifies the owner of the storage group.

**DATACLAS** `name`

This option specifies the name of the SMS data class.

**NODATACLAS**

This option removes an existing data class.

**MGMTCLAS** `name`

This option specifies the name of the SMS management class.
NOMGMTCLAS

This option removes an existing management class.

STORCLAS name

This option specifies the name of the SMS storage class.

NOSTORCLAS

This option removes an existing storage class.

CREATE SYNONYM statement

The CREATE SYNONYM statement creates a synonym for an existing table.

Figure 190: CREATE SYNONYM statement

CREATE SYNONYM synonynOwner.synonymName

This option specifies the fully qualified name of the synonym to be created.

TABLE

This option specifies the new table or view that the synonym references. Table 221 on page 516 lists valid values for the TABLE parameter.

Table 221: TABLE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableOwner.tableName</td>
<td>Specifies the name of a table</td>
</tr>
<tr>
<td>viewOwner.viewName</td>
<td>Specifies the name of a view</td>
</tr>
</tbody>
</table>

CREATE TABLE statement

The CREATE TABLE statement defines a new persistent table or global temporary table to the DB2 system.
Note
All of the parameters apply only to persistent tables unless otherwise noted.

Figure 191: CREATE TABLE statement (page 1)

CREATE TABLE

CREATE TABLE  `tableOwner1`.`tableName1`  
  ,  
  SQLID SQLID

COLUMNS  (  columnDefinition  )

PERIOD  BUSINESS_TIME( `columnName1` , `columnName2` )  EXCLUSIVE

PERIOD SYSTEM_TIME( `columnName3` , `columnName4` )  INCLUSIVE

ARCHIVING_TABLE  `tableOwner2`.`tableName2`

VERSIONING_TABLE  `tableOwner3`.`tableName3`

CLONE  NO

YES

REFERENCETB  `tableOwner4`.`tableName4`

MQT  NO

YES  MQTParameters

DATABASE  `databaseName`

TABLESPACE  `tableSpaceName`

<TDEFLT>

TYPE  TEMPORARY

PERSISTENT

EDITPROC  `procedureName`

VALIDPROC  `procedureName`

AUDIT  NONE

CHANGES

ALL

OBID  `integer`

NUMPARTS  `number`

(PART  `number`  `tablePartitionParameters`)

CREATE TABLE page 2
CREATE TABLE `tableOwner`.`tableName`

This option specifies the fully qualified name of the table to be created.

SQLID SQLID

This option specifies the current SQLID.

COLUMNS (columnDefinition, ...)

This option specifies the column definitions for the table according to `columnDefinition` (see “columnDefinition” on page 525).

PERIOD BUSINESS_TIME(`columnName 1`, `columnName 2`)

This option specifies the BUSINESS_TIME period for the table. `columnName 1` identifies the column that begins the time period. `columnName 2` identifies the column that ends the time period.

Table 222 on page 518 lists valid values for the BUSINESS_TIME parameter (valid for DB2 Version 12.1 and later).

Table 222: BUSINESS_TIME parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXCLUSIVE</td>
<td>The beginning value is included in the period, and the ending value is excluded.</td>
</tr>
<tr>
<td>INCLUSIVE</td>
<td>The beginning and ending values are both included in the period.</td>
</tr>
</tbody>
</table>

PERIOD SYSTEM_TIME(`columnName 3`, `columnName 4`)

This option specifies the SYSTEM_TIME period for the table. `columnName 3` identifies the column that begins the time period. `columnName 4` identifies the column that ends the time period.

VERSIONING_TABLE `tableOwner`.`tableName`

This option specifies the name of the history table.

ARCHIVING_TABLE `tableOwner`.`tableName`

This option specifies the name of the archive table.

CLONE

This option specifies the clone table. Table 223 on page 519 lists valid values for the CLONE parameter.
Table 223: CLONE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that the table should not be defined as a clone table</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies that the table should be defined as a clone table</td>
</tr>
</tbody>
</table>

**REFERENCETB tableOwner 5,tableName 6**

This option specifies the name of the table to which the clone table references.

**MQT**

This option specifies the materialized query table (MQT). Table 224 on page 519 lists valid values for the MQT parameter.

Table 224: MQT parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that the table should not be defined as a materialized query table</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies that the table should be defined as a materialized query table</td>
</tr>
</tbody>
</table>

**MQTParameters**

This option specifies the parameters for the MQT (see “MQTParameters” on page 532).

**DATABASE databaseName**

This option specifies the database for the table.

**TABLESPACE**

This option specifies the new table space for the table. Table 225 on page 519 lists valid values for the TABLESPACE parameter.

Table 225: TABLESPACE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableSpaceName</td>
<td>Specifies the name of the table space</td>
</tr>
</tbody>
</table>
| <DEFLT>   | Specifies a default value for an implicit table space  
|           | <DEFLT> is a reserved keyword. |

**TYPE**

This option specifies the type of table to create. Table 226 on page 520 lists valid values for the TYPE parameter.
### Table 226: TYPE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEMPORARY</td>
<td>Specifies that the table will be a global temporary table</td>
</tr>
<tr>
<td>PERSISTENT</td>
<td>Denotes that a standard DB2 table will be created</td>
</tr>
</tbody>
</table>

**EDITPROC** `procedureName`

This option specifies an edit procedure for the table.

**VALIDPROC** `procedureName`

This option specifies a validation procedure for the table.

**AUDIT**

This option specifies the audit procedure for the table. Table 227 on page 520 lists valid values for the AUDIT parameter.

### Table 227: AUDIT parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>Specifies not to perform auditing</td>
</tr>
<tr>
<td>CHANGES</td>
<td>Specifies to perform auditing when the table is updated</td>
</tr>
<tr>
<td>ALL</td>
<td>Specifies to perform auditing when any action is performed</td>
</tr>
</tbody>
</table>

**OBID** `integer`

This option specifies the value of the internal identifier of a table (OBID) in a read-only shared database.

**NUMPARTS** `number (PART number tablePartitionParameters)`

This option defines the new number of partitions for the table. For a partitioned table, NUMPARTS defines the parameters for PART `number` according to `tablePartitionParameters` (see “`tablePartitionParameters`” on page 533).

**DATA CAPTURE**

This option specifies whether additional information is added to the logging of SQL commands. Table 228 on page 521 lists valid values for the DATA CAPTURE parameter.
Table 228: DATA CAPTURE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>Specifies not to include additional information</td>
</tr>
<tr>
<td>CHANGES</td>
<td>Specifies to write additional information to the log</td>
</tr>
</tbody>
</table>

**VOLATILE**

This option specifies whether DB2 should use index access to the table whenever possible. Table 229 on page 521 lists valid values for the VOLATILE parameter.

Table 229: VOLATILE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies that SQL access to the table should be based on current statistics</td>
</tr>
</tbody>
</table>
Table 230: DROPRESTRICT parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to restrict the table</td>
</tr>
</tbody>
</table>
Value | Description
---|---
YES | Specifies to restrict the table
  The parent table space and database of a table with DROPRESTRICT YES cannot be dropped.

**APPEND**

This option specifies whether to use append processing for the table. With append processing, DB2 attempts to place table rows in accordance with the value in the rows’ cluster key columns when inserting or loading data. Table 231 on page 523 lists valid values for the APPEND parameter.

**Table 231: APPEND parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to use append processing</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to use append processing</td>
</tr>
</tbody>
</table>

**CCSID**

*(global temporary tables)* This option specifies the new encoding scheme of the table. Table 232 on page 523 lists valid values for the CCSID parameter.

**Table 232: CCSID parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>Specifies ASCII encoding</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>Specifies EBCDIC encoding</td>
</tr>
<tr>
<td>UNICODE</td>
<td>Specifies Unicode encoding</td>
</tr>
</tbody>
</table>

**PRIMARYKEY (columnName, ...)**

This option specifies the primary key definition for the table.

**KEYCOLUMNS (columnName order, ...)**

This option specifies the names of the partitioning key columns for the table. Table 233 on page 523 lists valid values for the order.

**Table 233: order values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASC (default value)</td>
<td>Places the columns in ascending order</td>
</tr>
<tr>
<td>DESC</td>
<td>Places the columns in descending order</td>
</tr>
</tbody>
</table>
COMMENT 'string'

This option defines a comment for the table.

LABEL 'string'

This option defines a label for the table.

COLCOMMENT columnName 'string'

This option defines a comment 'string' on columnName.

COLLABEL columnName 'string'

This option defines a label 'string' on columnName.
**columnDefinition**

*columnDefinition* uses the following parameters.

Figure 193: *columnDefinition*

```
columnDefinition
  ( columnName
    builtinDataType
      TYPESCHEMA
      - dataTypeSchema
      - TYPENAME
    )
```  

- **FOR**: BIT, DATA
  - SBCS
  - MIXED
- **FIELDPROC**: `procedureName`
- **INLINE LENGTH**: `integer`
- **nullParameters**
  - **IDENTITY**: NO, YES
    - START: `decimal`
    - INCREMENT: `integer`
    - CACHE: `integer`
    - ORDER: YES
    - CYCLE: YES
    - MAXVALUE: `decimal`
    - MINVALUE: `decimal`
- **ROWCHANGE**: NO, YES
- **HIDDEN**: NO, YES

**columnName**

*(persistent and global temporary tables)* This option specifies the type of column. Table 234 on page 526 lists valid values for the *columnName*.
### Table 234: columnName values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>builtInDataType</td>
<td>Defines the column as the specified DB2 data type (see Figure 194 on page 526)</td>
</tr>
<tr>
<td>TYPESCHEMA dataTypeSchema</td>
<td>Specifies the schema type of the column</td>
</tr>
<tr>
<td>TYPENAME dataTypeName</td>
<td>Specifies the distinct type of the column</td>
</tr>
</tbody>
</table>

#### Figure 194: builtInDataType (page 1)

_builtInDataType_

![builtInDataType diagram](image-url)
For information about valid data types, see the IBM documentation.

**FOR subtype DATA**

*(persistent and global temporary tables)* For character (CHAR), varying-length character (VARCHAR), and character large object (CLOB) columns, defines the character subtype. *Table 235 on page 527* lists valid values for the *subtype*.

**Note**

If the parent table space or database is defined as Unicode, and the column is created with an implicit subtype, the FOR *subtype* DATA parameter is omitted from the column definition.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIT</td>
<td>Defines the subtype as a binary character set</td>
</tr>
<tr>
<td>SBCS</td>
<td>Defines the subtype as a single-byte character set</td>
</tr>
</tbody>
</table>
FIELDPROC `procedureName (constant, ...)`

This option defines a field procedure for the defined column.

INLINE LENGTH `integer`

This option specifies the length for an inline LOB column. A value of 0 indicates that the column is not a LOB column, or that the LOB column does not have a length.

nullParameters

Provides additional options for null parameters (see “nullParameters” on page 443).

IDENTITY

Indicates whether the table includes an identity column. Each table can have one identity column. Table 236 on page 528 lists valid values for the IDENTIFY parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that the table does not include an identity column.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that the table includes an identity column.</td>
</tr>
</tbody>
</table>

If the table includes an identity column, the parameters in Table 237 on page 528 describe the column.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| START `decimal` | Specifies the starting value for the identity column  
The default value is the minimum value for a column in ascending sequence or the maximum value for a column in descending sequence. |
| INCREMENT `integer` | Specifies the amount of change in the values for the identity column  
The value can be any positive or negative integer, including 0 (zero). The default value is 1. |
| CACHE `integer` | Specifies the number of a set of values that are stored for the identity column  
A value of 0 (zero) is the equivalent of NO CACHE. |
Value | Description
--- | ---
**CYCLE** | Indicates whether values should be generated after the maximum or minimum value is reached  
- **NO** (default) indicates that values should not be generated after the maximum or minimum value is reached  
- **YES** indicates that values should be generated after the maximum or minimum value is reached

**ORDER** | Indicates whether values must be generated in the order in which they are requested  
- **NO** (default) indicates that values should not be generated in the order in which they are requested  
- **YES** indicates that values must be generated in the order in which they are requested

**MAXVALUE decimal** | Specifies the maximum value for an identity column  
The maximum value can be equal to or greater than the minimum value.

**MINVALUE decimal** | Specifies the minimum value for an identity column  
The minimum value can be equal to or less than the maximum value.

**ROWCHANGE**

Indicates whether to generate the time stamp of the most recent change to a row in a column. The data type must be TIMESTAMP, and the null parameter must be NOT NULL. Table 238 on page 529 lists valid values for the ROWCHANGE parameter.

**Table 238: ROWCHANGE parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that a value should not be generated for the time stamp</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that a value should be generated for the time stamp</td>
</tr>
</tbody>
</table>

**HIDDEN**

Indicates whether the column can be "hidden" or excluded from the results of a SELECT * SQL statement. Table 239 on page 530 lists valid values for the HIDDEN parameter.
Table 239: HIDDEN parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that the column is visible in the results of the statement</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that the column is hidden from the results of the statement</td>
</tr>
<tr>
<td></td>
<td>You cannot specify HIDDEN YES for a ROWID column.</td>
</tr>
</tbody>
</table>

nullParameters

nullParameters uses the following parameters.

Figure 196: nullParameters

```
nullParameters
```

NOT NULL

This option specifies that the column cannot contain any null values. Table 240 on page 530 lists valid values for the NOT NULL parameter.

Table 240: NOT NULL parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WITH DEFAULT</td>
<td>Specifies that the column has a default value</td>
</tr>
<tr>
<td></td>
<td>If a default value is not specified, CHANGE MANAGER uses a value assigned by</td>
</tr>
<tr>
<td></td>
<td>the system.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| DEFAULTVALUE     | Specifies the default value of the column:  
|                  | - *constant* specifies a numeric or non-numeric constant as the default value  
|                  | - USER specifies the value of the CURRENT USER special register  
|                  | - CURRENT SQLID specifies the value of the CURRENT SQLID special register |
| GENERATED        | For an identity column or a ROWID column, indicates when the value for the column is generated by DB2:  
|                  | - ALWAYS indicates that the value for the column is generated by DB2  
|                  | - DEFAULT indicates that the value for the column is generated by DB2 when a value is not specified |

**NULL**

This option specifies that the column can contain any null values. Any column added to a global temporary table must have a default value of null. Table 241 on page 531 lists valid values for the NULL parameter.

**Table 241: NULL parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| DEFAULT        | Specifies that the column has a default value  
|                | If a default value is not specified, CHANGE MANAGER uses a value assigned by the system. |
| DEFAULTVALUE   | Specifies the default value of the column:  
|                | - *constant* specifies a numeric or non-numeric constant as the default value  
|                | - USER specifies the value of the CURRENT USER special register  
|                | - CURRENT SQLID specifies the value of the CURRENT SQLID special register |
**MQTParameters**

MQTParameters uses the following parameters for a materialized query table.

**Figure 197: MQTParameters**

MQTParameters

- **PATH (schemaList)**
  
  This option specifies the path for the materialized query table.

- **TEXT (MQTText)**
  
  This option specifies the text for the materialized query table.

- **AS SELECT subselect**
  
  This option defines the SQL subselect statement that is used to create the materialized query table. The AS SELECT clause is required.

- **MAINTAINED BY**
  
  This option specifies whether the system or the user maintains data in the materialized query table. Table 242 on page 532 lists valid values for the MAINTAINED BY parameter.

**Table 242: MAINTAINED BY parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>Specifies that the system maintains the materialized query table</td>
</tr>
<tr>
<td>USER</td>
<td>Specifies that the user maintains the materialized query table</td>
</tr>
</tbody>
</table>
QUERYOPT

This option specifies whether the materialized query table can be used to optimize queries. Table 243 on page 533 lists valid values for the QUERYOPT parameter.

Table 243: QUERYOPT parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies that the table can be used to optimize queries (in other words, enables optimization)</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies that the table cannot be used to optimize queries (in other words, disables optimization)</td>
</tr>
</tbody>
</table>

$tablePartitionParameters$

$tablePartitionParameters$ uses the following parameter.

Figure 198: $tablePartitionParameters$

VALUES (constant, ...)

This option specifies the limiting constants for this partition of the table.

CREATE TABLESPACE statement

The CREATE TABLESPACE statement defines a new table space to a DB2 system.
**Figure 199: CREATE TABLESPACE statement**

```
CREATE TABLESPACE databaseName.tableSpaceName
```

This option specifies the fully qualified name of the table space to be created.

**LARGE**

This option indicates whether the table space is a LARGE table space. Table 244 on page 534 lists valid values for the LARGE parameter.

**Table 244: LARGE parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>For a partitioned table space, indicates that each partition has a maximum partition size of 4 GB</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that the table space is not a LARGE table space</td>
</tr>
</tbody>
</table>
LOB

This option indicates whether the table space is a nonpartitioned table space that contains the data for a large object (LOB) column in an auxiliary table. Table 245 on page 535 lists valid values for the LOB parameter.

Table 245: LOB parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that the table space is not a nonpartitioned table space that contains LOB data</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that the table space is a nonpartitioned table space that contains LOB data</td>
</tr>
</tbody>
</table>

OWNER authorizationID

This option specifies the owner of the table space.

DSSIZE integer G

This option specifies the maximum size for each table space partition in gigabytes.

MEMBER CLUSTER

This option indicates whether inserted data is clustered by the clustering index. Table 246 on page 535 lists valid values for the MEMBER CLUSTER parameter.

Table 246: MEMBER CLUSTER parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that data is clustered based on the availability of space map pages</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that data is not clustered by the clustering index</td>
</tr>
</tbody>
</table>

NUMPARTS number

This option specifies the number of partitions for a table space. NUMPARTS 0 specifies a nonpartitioned table space.

NUMPARTS number PART number tableSpacePartitionParameters

This option specifies each partition of the table space according to the partition number and tableSpacePartitionParameters (see “tableSpacePartitionParameters” on page 537). If PART 64 (or greater) is specified, a LARGE table space is created, even if the LARGE keyword is not specified.
PART 0 `tableSpacePartitionParameters`

This option specifies a nonpartitioned table space according to `tableSpacePartitionParameters` (see “`tableSpacePartitionParameters`” on page 537).

LOCKPART

This option indicates whether selective partition locking is used when locking a partitioned table space. Table 247 on page 536 lists valid values for the LOCKPART parameter.

Table 247: LOCKPART parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Indicates that selective partition locking is not used. LOCKPART NO is the default.</td>
</tr>
<tr>
<td>YES</td>
<td>Indicates that selective partition locking is used.</td>
</tr>
</tbody>
</table>
**tableSpacePartitionParameters**

tableSpacePartitionParameters uses the following parameters.

![Diagram of tableSpacePartitionParameters]

**VCAT catalogName**

This option specifies the volume catalog to be used for the table space.

**STOGROUP**

This option specifies the new default storage group for the table space. Table 248 on page 537 lists valid values for the STOGROUP parameter.

**Table 248: STOGROUP parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>storageGroupName</td>
<td>Specifies the name of the storage group</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value &lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>
PRIQTY number
Specifies the primary space that is allocated for the DB2-defined data set.

SECQTY number
Specifies the secondary space that is allocated for the DB2-defined data set.

ERASE
Indicates whether the data sets can be erased when they are deleted by a statement that drops the table space.

- NO (default)
  Specifies not to erase the data sets
- YES
  Specifies to erase the data sets

FREEPAGE number
This option specifies that a free page is added after every number of pages upon table space creation or reorganization.

PCTFREE number
This option specifies the percentage of free space left on each page of the table space upon creation or reorganization.

TRACKMOD
This option indicates whether modified pages are tracked in the space map pages of the table space. Table 249 on page 538 lists valid values for the TRACKMOD parameter.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to track modified pages TRACKMOD YES is the default.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies not to track modified pages</td>
</tr>
</tbody>
</table>

COMPRESS
This option indicates whether rows in the table space or partition are compressed. Table 250 on page 539 lists valid values for the COMPRESS parameter.
**Table 250: COMPRESS parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>Specifies not to compress data</td>
</tr>
<tr>
<td></td>
<td>COMPRESS NO is the default.</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies to compress data</td>
</tr>
</tbody>
</table>

**GBPCACHE**

This option specifies which pages are cached to the group buffer pool for this partition. Table 251 on page 539 lists valid values for the GBPCACHE parameter.

**Table 251: GBPCACHE parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANGED</td>
<td>Specifies to cache updated pages to the buffer pool</td>
</tr>
<tr>
<td></td>
<td>GBPCACHE CHANGED is the default.</td>
</tr>
<tr>
<td>ALL</td>
<td>Specifies to cache all pages as they are read</td>
</tr>
<tr>
<td>NONE</td>
<td>Specifies not to cache pages to the buffer pool</td>
</tr>
</tbody>
</table>
tableSpaceOptions

tableSpaceOptions includes the following parameters.

Figure 201: tableSpaceOptions

```
BUFFERPOOL  bufferPoolName
             <DEFLT>

LOCKSIZE
    ANY
    PAGE
    TABLE
    TABLESPACE
    ROW
    LOB

LOCKMAX  integer
         SYSTEM

CLOSE
    YES
    NO

MAXPARTITIONS  integer

CCSID
    ASCII
    EBCDIC
    UNICODE
    <DEFLT>

MAXROWS  integer

LOG
    YES
    NO

DEFINE
    YES
    NO

SEGSIZE  number
```

**BUFFERPOOL**

This option specifies the new default buffer pool for the database for the creation of the table space. If the BUFFERPOOL keyword is not specified, the DB2 default value is used. Table 252 on page 541 lists valid values for the BUFFERPOOL parameter.
Table 252: BUFFERPOOL parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufferPoolName</td>
<td>Specifies the name of the buffer pool</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value</td>
</tr>
<tr>
<td></td>
<td>&lt;DEFLT&gt; is a reserved keyword.</td>
</tr>
</tbody>
</table>

LOCKSIZE

This option specifies the size of the locks for the table space. Table 253 on page 541 lists valid values for the LOCKSIZE parameter.

Table 253: LOCKSIZE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANY</td>
<td>Specifies to use any lock size</td>
</tr>
<tr>
<td>PAGE</td>
<td>Specifies to use page locks</td>
</tr>
<tr>
<td>TABLE</td>
<td>Specifies to use table locks</td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>Specifies to use table space locks</td>
</tr>
<tr>
<td>ROW</td>
<td>Specifies to use row locks</td>
</tr>
<tr>
<td>LOB</td>
<td>For LOB table spaces, specifies to use LOB locks</td>
</tr>
</tbody>
</table>

LOCKMAX

This option specifies the maximum number of locks for the table space. Table 254 on page 541 lists valid values for the LOCKMAX parameter.

Table 254: LOCKMAX parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>integer</td>
<td>Specifies the number of page, row, or LOB locks</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>Specifies the value used by DB2 for the number of locks</td>
</tr>
</tbody>
</table>

CLOSE

This option indicates whether the index for the table space can be closed when the index is not being used. Table 255 on page 542 lists valid values for the CLOSE parameter.
Table 255: CLOSE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies that the index can be closed</td>
</tr>
<tr>
<td></td>
<td>CLOSE YES is the default.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies that the index cannot be closed</td>
</tr>
</tbody>
</table>

**MAXPARTITIONS** \(\text{integer}\)

This option specifies the maximum number of partitions in a partition-by-growth table space. The valid range of values is 1 through 4096.

**CCSID**

This option specifies the encoding scheme of the table space. Table 256 on page 542 lists valid values for the CCSID parameter.

Table 256: CCSID parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>Specifies ASCII encoding</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>Specifies EBCDIC encoding</td>
</tr>
<tr>
<td>UNICODE</td>
<td>Specifies Unicode encoding</td>
</tr>
<tr>
<td>&lt;DEFLT&gt;</td>
<td>Specifies a default value</td>
</tr>
<tr>
<td></td>
<td>The default encoding scheme is determined in the installation options module with the CCSID keyword.</td>
</tr>
</tbody>
</table>

**MAXROWS** \(\text{integer}\)

This option specifies the maximum number of rows per page for the table space. The valid range of values is 1 through 255. This parameter does not apply to LOB table spaces.

**LOG**

This option indicates whether changes to data in a table space are written to a log. Table 257 on page 542 lists valid values for the LOG parameter.

Table 257: LOG parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Specifies to write changes to a log</td>
</tr>
<tr>
<td></td>
<td>LOG YES is the default.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies not to write changes to a log</td>
</tr>
</tbody>
</table>
DEFINE

This option indicates when the data sets for the table space are created. Table 258 on page 543 lists valid values for the DEFINE parameter.

**Table 258: DEFINE parameter values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that DB2 creates the data sets when it creates the table space. DEFINE YES is the default.</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that DB2 creates the data sets when a row is inserted into a table</td>
</tr>
</tbody>
</table>

**SEGSIZE number**

This option specifies the number of pages for each segment of a segmented table space. SEGSIZE 0 specifies an unsegmented table space.

**Note**

DB2 requires SEGSIZE to be between 4 and 64 pages and divisible by 4.

**CREATE TRIGGER statement**

The CREATE TRIGGER statement defines a new trigger.
Figure 202: CREATE TRIGGER statement

CREATE TRIGGER

This option specifies the fully qualified name of the trigger to be created.

OWNER triggerOwner

This option specifies the owner of the trigger.

TABLE tableOwner.tableName

This option specifies the triggering base table of the associated trigger.
TBQUALIFIED

This option indicates whether the TABLE name was qualified in the CREATE TRIGGER statement. Table 259 on page 545 lists valid values for the TBQUALIFIED parameter.

Table 259: TBQUALIFIED parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that the TABLE name was qualified</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that the TABLE name was not qualified</td>
</tr>
</tbody>
</table>

ACTIVATE

This option specifies whether the trigger is activated before or after the triggering event. Table 260 on page 545 lists valid values for the ACTIVATE parameter.

Table 260: ACTIVATE parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BEFORE</td>
<td>Specifies to activate the trigger before the event</td>
</tr>
<tr>
<td>AFTER</td>
<td>Specifies to activate the trigger after the event</td>
</tr>
</tbody>
</table>

GRANULARITY

This option specifies whether DB2 executes the triggered action for each row or for each statement. Table 261 on page 545 lists valid values for the GRANULARITY parameter.

Table 261: GRANULARITY parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROW</td>
<td>Specifies to execute the action for each row</td>
</tr>
<tr>
<td>STATEMENT</td>
<td>Specifies to execute the action for each statement</td>
</tr>
</tbody>
</table>

SECURED

This option specifies that the trigger is secured.

NOT SECURED

This option specifies that the trigger is not secured.
ONEVENT

This option specifies when DB2 executes the trigger. Table 262 on page 546 lists valid values for the ONEVENT parameter.

Table 262: ONEVENT parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSERT</td>
<td>Specifies to have DB2 execute the trigger when an insertion occurs</td>
</tr>
<tr>
<td>DELETE</td>
<td>Specifies to have DB2 execute the trigger when a deletion occurs</td>
</tr>
<tr>
<td>UPDATE ALLCOLUMNS</td>
<td>Specifies to have DB2 execute the trigger when an update occurs, and to update all of the columns in the table</td>
</tr>
<tr>
<td>UPDATE COLUMNS columnNameList</td>
<td>Specifies to have DB2 execute the trigger when an update occurs, and to update a list of columns in columnNameList</td>
</tr>
</tbody>
</table>

REFOLDCORR correlationName

This option specifies the OLD AS correlation name.

REFNEWCORR correlationName

This option specifies the NEW AS correlation name.

REFOLDTB identifier

This option specifies the OLD _TABLE AS identifier.

REFNEWTB identifier

This option specifies the NEW_TABLE AS identifier.

COMMENT 'string'

This option specifies the comment for the trigger.

PATH (schemaList)

This option specifies the path for the trigger.

TRIGGERTEXT (triggerText)

This option specifies the triggered action for the trigger.
CREATE VIEW statement

The CREATE VIEW statement defines a new view.

Figure 203: CREATE VIEW statement

CREATE VIEW

CREATE VIEW viewOwner . viewName SQLID SQLID

QUALIFIED YES NO

CHECK OPTION LOCAL YES NO

COLUMNS ( columnName )

COMMENT 'string'

LABEL 'string'

\ startRow \ COLCOMMENT columnName 'string'

\ startRow \ COLLABEL columnName 'string'

PATH ( schemaList )

AS SELECT subselect

CREATE VIEW viewOwner . viewName

This option specifies the fully qualified name of the view being created.

SQLID SQLID

This option specifies the current SQLID.

QUALIFIED

This option indicates whether the view was created with a qualified name. Table 263 on page 548 lists valid values for the QUALIFIED parameter.
Table 263: QUALIFIED parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>Indicates that the view was created with a qualified name</td>
</tr>
<tr>
<td>NO</td>
<td>Indicates that the view was not created with a qualified name</td>
</tr>
</tbody>
</table>

CHECK OPTION

This option specifies the new check option for the view. Views that reference global temporary tables cannot specify CHECK OPTION. Table 264 on page 548 lists valid values for the CHECK OPTION parameter.

Table 264: CHECK OPTION parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCAL</td>
<td>Specifies that all search conditions are checked conditionally</td>
</tr>
<tr>
<td>YES</td>
<td>Specifies that all search conditions are checked, regardless of the check options specified CHECK OPTION YES creates a cascaded view.</td>
</tr>
<tr>
<td>NO</td>
<td>Specifies not to use the definition of the view to check operations that use the view</td>
</tr>
</tbody>
</table>

COLUMNS (columnName, ...)

This option specifies the names of the columns to be included in the view.

COMMENT 'string'

This option defines a comment for the view.

LABEL 'string'

This option defines a label for the view.

COLCOMMENT columnName 'string'

This option defines a comment 'string' on columnName.

COLLABEL columnName 'string'

This option defines a label 'string' on columnName.

PATH (schemaList)

This option specifies the path for this view.
AS SELECT subselect

This option defines a new SQL subselect statement as the definition for the view. The AS SELECT clause is required and must appear last in the CREATE VIEW statement.

**DROP ALIAS statement**

The DROP ALIAS statement removes an alias definition.

**Figure 204: DROP ALIAS statement**

```
DROP ALIAS
```

To drop an alias, you must have one of the following privileges:

- Ownership of the object
- SYSADM or SYSCTRL authority

DROP ALIAS aliasOwner. aliasName

This option specifies the fully qualified name of the alias to be dropped.

**DROP CHECK statement**

The DROP CHECK statement removes a check constraint from a table.

Compare builds this form of a DROP CHECK statement when CMPIN1 contains an unnamed check constraint.

**Figure 205: DROP CHECK statement**

```
DROP CHECK
```

This option specifies the fully qualified name of the constraint to be dropped.
**DROP CHECK** `tableOwner$tableName.constraintName`

This option specifies the name of the constraint to be dropped.

**DROP CHECK** `tableOwner$tableName.CHECK (constraintText)`

This option specifies the name of the constraint to be dropped and the check constraint text.

---

**DROP UNIQUE CONSTRAINT statement**

The DROP UNIQUE CONSTRAINT statement removes a unique constraint from a table.

**Figure 206: DROP UNIQUE CONSTRAINT statement**

**DROP UNIQUE CONSTRAINT**

<table>
<thead>
<tr>
<th>DROP UNIQUE CONSTRAINT</th>
<th><code>tableOwner</code></th>
<th><code>tableName</code></th>
<th><code>uniqueConstraintName</code></th>
</tr>
</thead>
</table>

This option specifies the fully qualified name of the unique constraint to be dropped.

**DROP DATABASE statement**

The DROP DATABASE statement removes a database and all of its dependent objects.

**Figure 207: DROP DATABASE statement**

**DROP DATABASE**

<table>
<thead>
<tr>
<th>DROP DATABASE</th>
<th><code>databaseName</code></th>
</tr>
</thead>
</table>

To drop a database, you must have one of the following privileges:

- The DROP privilege on the database
- DBADM or DBCTRL authority for the database
- SYSADM or SYSCTRL authority

**Note**
You cannot use a DROP DATABASE statement to implicitly drop a global temporary table. Global temporary tables must be dropped in DROP TABLE statements.

DROP DATABASE *databaseName*

This option specifies the name of the database to be dropped.

**DROP FOREIGN KEY statement**

The DROP FOREIGN KEY statement removes a referential integrity constraint from a table.

*Figure 208: DROP FOREIGN KEY statement*

**DROP FOREIGN KEY**

DROP FOREIGN KEY  \( \text{tableOwner}_1 \cdot \ldots \cdot \text{tableName}_1 \cdot \ldots \cdot \text{constraintName} \)

**REFERENCETB**  \( \text{tableOwner}_2 \cdot \ldots \cdot \text{tableName}_2 \)

**KEYCOLUMNS** ( \( \text{columnName}, \ldots \) )

**REFCOLUMNS** ( \( \text{columnName}, \ldots \) )

**DROP FOREIGN KEY** \( \text{tableOwner}_1.\text{tableName}_1.\text{constraintName} \)

This option specifies the fully qualified name of the constraint to be dropped.

**REFERENCETB** \( \text{tableOwner}_2.\text{tableName}_2 \)

This option specifies the name of the table to which the foreign key references.

**KEYCOLUMNS** ( \( \text{columnName}, \ldots \) )

This option specifies the list of columns that the foreign key references.
REFCOLUMNS (columnName, ...)

This option specifies the names of the parent key columns.

**DROP INDEX statement**

The DROP INDEX statement removes an index from a table.

**Figure 209: DROP INDEX statement**

```
DROP INDEX indexOwner.indexName
```

To drop an index, you must have one of the following privileges:

- Ownership of the object (the owner of the table or the index)
- DBADM authority
- PACKADM authority for the collection or for all collections

**Note**

Partitioned indexes can only be dropped with a DROP TABLESPACE statement on the associated table space.

If a unique index is dropped and that index was used to enforce the uniqueness of a parent key, the definition of the parent table is changed to incomplete.

```
DROP INDEX indexOwner.indexName
```

This option specifies the name of the index to be dropped.

**DROP PROCEDURE statement**

The DROP PROCEDURE statement removes a stored procedure.

**Figure 210: DROP PROCEDURE statement**

```
DROP PROCEDURE procedureOwner - . - procedureSchema - . - procedureName
```
DROP PROCEDURE `procedureOwner`.`procedureSchema`. `procedureName`

This option specifies the fully qualified name of the stored procedure to be dropped.

**DROP SEQUENCE statement**

The DROP SEQUENCE statement removes a sequence.

*Figure 211: DROP SEQUENCE statement*

```
DROP SEQUENCE
```

`DROP SEQUENCE schema.sequenceName`

This option specifies the fully qualified name of the sequence to be dropped.

**DROP STOGROUP statement**

The DROP STOGROUP statement removes a storage group definition.

*Figure 212: DROP STOGROUP statement*

```
DROP STOGROUP
```

To drop a stogroup, you must have one of the following privileges:

- Ownership of the object
- SYSADM or SYSCTRL authority

`DROP STOGROUP storageGroupName`

This option specifies the name of the storage group to be dropped.
DROP SYNONYM statement

The DROP SYNONYM statement removes a synonym definition.

Figure 213: DROP SYNONYM statement

```
DROP SYNONYM synonymOwner.synonymName
```

To drop a synonym, you must have ownership privileges for the synonym.

This option specifies the fully qualified name of the synonym to be dropped.

DROP TABLE statement

The DROP TABLE statement removes a persistent or global temporary table definition.

Figure 214: DROP TABLE statement

```
DROP TABLE tableOwner.tableName
```

To drop a table, you must have one of the following privileges:

- Ownership of the object
- DBADM authority
- PACKADM authority for the collection or for all collections

This option specifies the fully qualified name of the table to be dropped.
**DROP TABLESPACE statement**

The DROP TABLESPACE statement removes a table space and all of its dependent objects.

*Figure 215: DROP TABLESPACE statement*

```
DROP TABLESPACE
        databaseName . tableSpaceName
```

To drop a table, you must have one of the following privileges:

- Ownership of the object
- DBADM authority
- PACKADM authority for the collection or for all collections

**DROP TABLESPACE** *databaseName. tableSpaceName*

This option specifies the fully qualified name of the table space to be dropped.

**DROP TRIGGER statement**

The DROP TRIGGER statement removes a trigger definition.

*Figure 216: DROP TRIGGER statement*

```
DROP TRIGGER
        schema . triggerName
```

To drop a trigger, you must have one of the following privileges:

- Ownership of the object
- DROPIN privilege for the schema or all schemas
- SYSADM or SYSCTRL authority
DROP TRIGGER `schema.triggerName`

This option specifies the fully qualified name of the trigger to be dropped.

**DROP VIEW statement**

The DROP VIEW statement removes a view definition.

**Figure 217: DROP VIEW statement**

```
DROP VIEW

viewOwner . viewName
```

To drop a view, you must have one of the following privileges:

- Ownership of the object
- SYSADM or SYSCTRL authority

DROP VIEW `viewOwner.viewName`

This option specifies the fully qualified name of the view to be dropped.
CM/PILOT Data Manipulation Language (DML)

One feature of the CM/PILOT component is the use of SQL-like data manipulation language (DML) statements. The panels for two of the CM/PILOT scripts help you create DML statements to update, delete, and migrate data structures. You can also modify the DML statements in an ISPF edit session provided by the panel.

Actions that you can perform by using DML

Using DML provides you with a function similar to the CHANGE MANAGER Specification component, except that you can change or migrate multiple objects in a single DML statement. You can also import DML statements into an alter-type work ID. After the statements are imported, CHANGE MANAGER creates entries in the CD tables as if they had been requested in the Specification component.

The data structure changes and migrations that you specify are accomplished by searching and updating the DB2 catalog tables. Updates to the catalog occur when the Execution component of CHANGE MANAGER processes the CHANGE MANAGER worklist, which contains standard DDL statements.

Note

If you use the CHANGE MANAGER catalog indirection feature, CM/PILOT uses the same catalog tables (actual or copies) that you specified for CHANGE MANAGER to use. When you use the Execution component to process the worklist, the actual catalog tables are accessed and updated. For more information about catalog indirection, see:

- Installation System Reference Manual
- Installation System Quick Start
- BMC Products and Solutions for DB2 Customization Guide
Table 265 on page 558 shows the actions that you can perform on each object by using DML.

### Table 265: Actions that you can perform using DML

<table>
<thead>
<tr>
<th>Object</th>
<th>Update</th>
<th>Delete</th>
<th>Like</th>
<th>Migrate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alias</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Check constraint</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Column (table and view)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Database</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Foreign key</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Field</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>N/A</td>
</tr>
<tr>
<td>Index</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Index key column</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Index partition</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Parm</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Relation</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Routine</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Sequence</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Storage group</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Synonym</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Table</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Table space</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Table space partition</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Trigger</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>View</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Volume</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

For more information, view the Quick Course "CM/PILOT Overview."

---

## DML statements

The -DML command in a CM/PILOT worklist contains DML statements.
DML statements are used to update, delete, like, migrate, and change objects. A DML statement can include keywords, a SET clause, and a WHERE clause. You can import DML statements and populate a CHANGE MANAGER work ID’s CD tables. For information about importing DML statements, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

**Note**

If commas and equal signs are shown, you must enter them as part of the syntax. A semicolon (;) ends each DML statement. For information about the conventions that are used in the diagrams, see “Syntax diagrams” on page 13.

---

**Keywords**

The CM/PILOT component creates keywords for each of the DML objects. Table 266 on page 559 lists the keywords and the DB2 catalog table to which the keyword corresponds. When you modify a DML statement, you can use the keywords as they are shown, singular forms of the keywords, or the actual DB2 catalog names. The keywords and names are resolved to the CAT2 synonyms that are used by CHANGE MANAGER to access the DB2 catalog. For information about catalog synonyms, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 1*.

**Table 266: DML keywords and DB2 catalog tables**

<table>
<thead>
<tr>
<th>Object</th>
<th>Keyword</th>
<th>DB2 catalog table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alias</td>
<td>ALIASES</td>
<td>SYSIBM.SYSTABLES</td>
</tr>
<tr>
<td>Check constraint</td>
<td>CHECKS</td>
<td>SYSIBM.SYSCHECKS</td>
</tr>
<tr>
<td>Column(table and view)</td>
<td>COLUMNS</td>
<td>SYSIBM.SYSCOLUMNS</td>
</tr>
<tr>
<td>Database</td>
<td>DATABASES</td>
<td>SYSIBM.SYSDATABASE</td>
</tr>
<tr>
<td>Field</td>
<td>FIELDS</td>
<td>SYSIBM.SYSFIELDS</td>
</tr>
<tr>
<td>Foreign key</td>
<td>FOREIGNKEYS</td>
<td>SYSIBM.SYSFOREIGNKEYS</td>
</tr>
<tr>
<td></td>
<td>RELKEY</td>
<td>SYSIBM.SYSRELS</td>
</tr>
<tr>
<td>Index</td>
<td>INDEXES</td>
<td>SYSIBM.SYSINDEXES</td>
</tr>
<tr>
<td>Index key</td>
<td>KEYCOLUMNS</td>
<td>SYSIBM.SYSKEYS</td>
</tr>
<tr>
<td>Index partition</td>
<td>INDEXPARTS</td>
<td>SYSIBM.SYSINDEXPART</td>
</tr>
<tr>
<td>Parm</td>
<td>PARMS</td>
<td>SYSIBM.SYSPARMS</td>
</tr>
<tr>
<td>Relation</td>
<td>RELS</td>
<td>SYSIBM.SYSRELS</td>
</tr>
<tr>
<td>Routine</td>
<td>ROUTINES</td>
<td>SYSIBM.SYSRoutines</td>
</tr>
<tr>
<td>Object</td>
<td>Keyword</td>
<td>DB2 catalog table</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Sequence</td>
<td>SEQUENCES</td>
<td>SYSIBM.SYSSEQUENCES</td>
</tr>
<tr>
<td>Storage group</td>
<td>STOGROUPS</td>
<td>SYSIBM.SYSSTOGROUP</td>
</tr>
<tr>
<td>Synonym</td>
<td>SYNONYMS</td>
<td>SYSIBM.SYSSYNONYMS</td>
</tr>
<tr>
<td>Table (includes Global Temporary Table)</td>
<td>TABLES</td>
<td>SYSIBM.SYSTABLES</td>
</tr>
<tr>
<td>Table space</td>
<td>TABLESPACES</td>
<td>SYSIBM.SYSTABLESPACE</td>
</tr>
<tr>
<td>Table space partition</td>
<td>TABLEPARTS</td>
<td>SYSIBM.SYSTABLEPART</td>
</tr>
<tr>
<td>Trigger</td>
<td>TRIGGERS</td>
<td>SYSIBM.SYSTRIGGERS</td>
</tr>
<tr>
<td>View</td>
<td>VIEWS</td>
<td>SYSIBM.SYSTABLES</td>
</tr>
<tr>
<td>Volume</td>
<td>VOLUMES</td>
<td>SYSIBM.SYSVOLUMES</td>
</tr>
</tbody>
</table>

- The TYPE column in the SYSIBM.SYSTABLES table contains the value V for views.
- The TYPE column in the SYSIBM.SYSTABLES table contains the value A for aliases.
- The TYPE column in the SYSIBM.SYSTABLES table contains the value T for tables.

**SET clause**

The SET clause of a DML statement specifies changes to the attributes of the data structures or to the data structures.

**SET attribute = value**

Sets an attribute of an object to a value.

**attribute**

Specifies the attribute that you want to set. For a list of the attributes that you can set, see “Attributes used in SET and WHERE clauses” on page 604. For more information about attributes, see the descriptions in the IBM documentation.

You can find the associated catalog table name in “Keywords” on page 559. The term attribute as used in CM/PILOT DML is synonymous with the column names of DB2 catalog tables.

For example, if you want details about the LOCATION attribute of aliases, you can follow these steps:
1. Look at “Keywords” on page 559 of this guide. Use the SYSIBM.SYSTABLES catalog table for updating aliases.

2. Look in the IBM documentation that references the SYSIBM.SYSTABLES catalog table. The documentation should include a description of the LOCATION column name, which is synonymous with the LOCATION attribute that is used in CM/PILOT DML.

**value**

Specifies the new value for the attribute that you are setting. The value can be a constant or an expression. If you need more space than an input field provides on a panel, edit the DML statement. For example, you can set a NAME attribute with either of the following values: 'ABCD' or 'PROD' CONCAT SUBSTR(NAME,5). You can set a numeric attribute such as PQTY, with any of the following values: 123, PQTY * 2, or SQTY * 4.

**WHERE clause**

The WHERE clause of a DML statement defines the data structures to change or migrate. This definition is also known as a scope. You can use the panels to help create a scope. You can also edit a DML statement to create or tailor a scope.

**WHERE searchCondition**

The search condition consists of one or more predicates. You can use any search condition that is valid in a WHERE clause of an SQL SELECT statement. For information, see the IBM documentation.

**DELETE statement**

The DELETE statement deletes data structures.

You can import the DELETE statement in a DML file. For information, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

*Figure 218: DELETE statement*

```
DELETE  objectType  WHERE  searchCondition  ;
```
DELETE *objectType*

Specifies the object type to be deleted. The *objectType* variable can be one of the following keywords:

- ALIASES
- CHECKS
- COLUMNS
- DATABASES
- FIELDS
- FOREIGNKEYS
- INDEXES
- INDEXPARTS
- KEYCOLUMNS
- PARMS
- RELS
- ROUTINES
- SEQUENCES
- STOGROUPS
- SYNONYMS
- TABLEPARTS
- TABLES
- TABLESPACES
- TRIGGERS
- VIEWS
- VOLUMES

---

**Note**

The keywords that are listed are created by the CM/PILOT component when it builds a DML statement. When modifying a DML statement, you can use the keywords as they are shown, singular forms of the keywords, or the actual DB2 catalog names.

WHERE *searchCondition*

The *searchCondition* specifies the criteria that you use to define the scope for the delete action.

For details of the WHERE clause, see “WHERE clause” on page 561.

**Examples**

The following examples illustrate the use of the DELETE statement.
Example
The following example illustrates how to delete table and view columns that are named ZIPCODE and have a length of 5 characters.

```
DELETE COLUMNS
    WHERE NAME = 'ZIPCODE' AND LENGTH = 5;
```

Example
The following example illustrates how to drop databases that use the OLDGROUP storage group.

```
DELETE DATABASES
    WHERE STGROUP = 'OLDGROUP';
```

Example
The following example uses the JOIN keyword and executes in less time. For more information, see “Performance enhancements with the JOIN keyword” on page 602.

```
DELETE STOGROUPS
    JOIN VOLUMES
    WHERE VOLID = 'DDDD' AND
        SGNAME = NAME AND
        SGCREATOR = CREATOR ;
```

DELETE SYNCURRENT statement

The DELETE SYNCURRENT statement deletes the sync table entries for one or more work IDs.

In the CHANGE MANAGER CD table, none of the other entries that are associated with the original work ID are deleted. A task ID can contain one or more DELETE SYNCURRENT statements.

Note
CHANGE MANAGER does not support importing the DELETE SYNCURRENT statement. For more information, see the chapter about importing DML statements in ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2.

Figure 219: DELETE SYNCURRENT statement

```
DELETE SYNCURRENT
    WHERE searchCondition ;
```
DELETE SYNTABLE WHERE *searchCondition*

The *searchCondition* specifies the criteria that you use to define the scope for the delete action. The search condition consists of one or more predicates. You can use any search condition that is valid in a WHERE clause of an SQL SELECT statement. For information, see the IBM documentation.

The following CD table column attributes from the CM_WORKID table are valid for the *searchCondition* variable:

- WKOWNER
- TYPE
- CRDATE
- MODDATE
- WLDATE
- IMPORTDATE
- WLSTATTIME
- WKNAME
- SECURE
- CRTIME
- MODTIME
- WLTIME
- IMPORTTIME
- STATUS
- CRAUTH
- MODAUTH
- WLAUTH
- IMPORTAUTH
- WLSTATDATE

These column attributes are used to delete the rows from the CM_SYNC table.

**Example**

The following example illustrates how to delete the sync table entries for a work ID.

```sql
DELETE SYNTABLE
WHERE
  STATUS = 'C'
  (WKOWNER LIKE '%CRJ'
   OR WKOWNER LIKE '%DEM')
  AND DATE(MODDATE) < DATE('2010-07-03');
```

A report of the deleted sync table entries is provided in the diagnostic output in the AEXPRINT file.
DELETE WORKID statement

The DELETE WORKID statement deletes one or more work IDs.

All of the entries in the CHANGE MANAGER CD table that are associated with the original work ID are also deleted. A task ID can contain one or more DELETE WORKID statements.

**Note**

CHANGE MANAGER does not support importing the DELETE WORKID statement. For more information, see the chapter about importing DML statements in *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

**WARNING**

Full-recovery baselines are associated with alter-type work IDs. If you delete a work ID that has an associated full-recovery baseline, the work ID for that baseline will become null.

Figure 220: DELETE WORKID statement

```
DELETE WORKID
WHERE searchCondition
```

DELETE WORKID WHERE searchCondition

The *searchCondition* specifies the criteria that you use to define the scope for the delete action. The search condition consists of one or more predicates. You can use any search condition that is valid in a WHERE clause of an SQL SELECT statement. For information, see the IBM documentation.

In the CM/PILOT panel for DELETE WORKID, you can choose from one or more of the following CD table column attributes for the *searchCondition* variable:

- WKOWNER
- TYPE
- CRDATE
- MODDATE
- WLDATE
- IMPORTDATE
- WLSTATTIME
- WKNAME
- SECURE
LIKE statement

The LIKE statement copies all specified objects and applies the specified changes to the objects.
You can import the LIKE statement in a DML file. For information, see the ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2.

Figure 221: LIKE statement

```
LIKE
LIKE objectType SET attribute = value
WHERE searchCondition ;
```

**LIKE objectType**

Specifies the object type to be created. The `objectType` variable can be one of the following keywords:

- ALIASES
- CHECKS
- COLUMNS
- DATABASES
- FIELDS
- FOREIGNKEYS
- INDEXES
- INDEXPARTS
- KEYCOLUMNS
- PARMS
- RELS
- ROUTINES
- SEQUENCES
- STOGROUPS
- SYNONYMS
- TABLEPARTS
- TABLES
- TABLESPACES
- TRIGGERS
- VIEWS
- VOLUMES
Note

The keywords that are listed are created by the CM/PILOT component when it builds a DML statement. When modifying a DML statement, you can use the keywords as they are shown, singular forms of the keywords, or the actual DB2 catalog names.

**SET attribute = value**

For details of the SET clause, see “SET clause” on page 560.

**WHERE searchCondition**

Specifies the criteria that you use to define the scope for the action.

For details of the WHERE clause, see “WHERE clause” on page 561.

**Examples**

---

**Example**

The following example illustrates how to copy an alias with the LIKE statement.

```
LIKE ALIASES
SET CREATOR='NEW'
WHERE CREATOR='OLD';
```

---

**Example**

The following example illustrates how to copy a trigger.

```
LIKE TRIGGERS
SET NAME='NEW_' CONCAT SUBSTR(NAME,1,100)
WHERE SCHEMA='DEM0323A';
```

**MIGRATE statement**

The MIGRATE statement selects data structures for migration within the same DB2 subsystem or to other DB2 subsystems.

If you want to use change rules, you can use a CHANGE MANAGER outbound migrate profile. You can also create CHANGE MANAGER worklists for migration to multiple locations. For more information about creating and using outbound migrate profiles, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*. 
You can import the MIGRATE statement in a DML file. For information, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

**Figure 222: MIGRATE statement**

```
MIGRATE
    objectType 1
```

Specifies the object to be migrated. The `objectType 1` variable can be one of the following keywords:

- ALIASES
- CHECKS
- COLUMNS
- DATABASES
- FOREIGNKEYS
- INDEXES
- INDXPARTS
- KEYCOLUMNS
- PARMS
- RELS
- ROUTINES
- SEQUENCES
- STOGROUPS
- SYNONYMS
- TABLEPARTS
- TABLES
- TABLESPACES
Note
The keywords that are listed in the syntax diagrams for object types are created by the CM/PILOT component when it builds a DML statement. When modifying a DML statement, you can use the keywords as they are shown, singular forms of the keywords, or the actual DB2 catalog names.

**INCLUDE**

Specifies the data, authorizations, and objects (parent or dependent) to migrate, based on the object type that the MIGRATE keyword specifies. This clause provides a function similar to the work ID migrate options of CHANGE MANAGER.

For example, if you want to migrate a database and include its table spaces, tables, indexes, and data, create a MIGRATE statement with an INCLUDE clause that specifies those objects. When you select the script to migrate data structures that have dependent objects, the product builds the INCLUDE dependency1, dependency2, ... clause for you. You can modify this clause by editing the DML.

Note
You can use the auxiliary object in the INCLUDE clause with the database, table space, or table objects.

**ALL**

Specifies to include all data, authorizations, and dependent objects.

**DATA**

Specifies to include data.

**AUTHORIZATIONS**

Specifies to include object authorizations.

**PARENTS**

Specifies to include parent objects for table spaces, tables, columns, indexes, or index key columns, as shown in Table 267 on page 570.

**Table 267: Parent objects**

<table>
<thead>
<tr>
<th>Object</th>
<th>Parent object(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table space</td>
<td>Database</td>
</tr>
</tbody>
</table>
### objectType 2

Specifies a dependent object to include. The `objectType 2` variable can be one or more of the following keywords:

- ALIASES
- AUXILIARY
- CHECKS
- CONSTRAINTS
- FOREIGNKEYS
- INDEXES
- SYNONYMS
- TABLES
- TABLESPACES
- TRIGGERS
- VIEWS

### SET attribute = value

For details of the SET clause, see “SET clause” on page 560.

### WHERE searchCondition

Specifies the criteria that you use to define the scope for the action.

For details of the WHERE clause, see “WHERE clause” on page 561.

### Examples

The following examples illustrate the use of the MIGRATE statement. For additional examples and tasks, see the `ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2`.
Example
The following example illustrates how to migrate a database and change the buffer pool.

```sql
MIGRATE DATABASES
INCLUDE DATA, TABLES, FOREIGNKEYS, INDEXES, VIEWS,
   TABLESPACES, AUTHORIZATIONS, SYNONYMS, ALIASES, CHECKS
SET BPOOL='BP32K'
   WHERE NAME LIKE='ALUQA5%';
```

Example
The following example illustrates how to migrate the DEM database and its table spaces, tables, indexes, and data.

```sql
MIGRATE DATABASES
INCLUDE TABLESPACES, TABLES, INDEXES, DATA
   WHERE NAME = 'DEM';
```

Example
The following example illustrates how to migrate databases that use the ARUSG049 storage group. Note that the dependent objects of the database are not migrated.

```sql
MIGRATE DATABASES
   WHERE STGROUP = 'ARUSG049';
```

Example
The following example illustrates how to migrate databases that use storage groups with names that begin with ARU. Note that the dependent objects of the database are not migrated.

```sql
MIGRATE DATABASES
   WHERE STGROUP LIKE 'ARU%';
```

Example
The following example illustrates how to migrate the ARUSG049 storage group.

```sql
MIGRATE STOGROUPS
   WHERE NAME = 'ARUSG049';
```

Example
The following example illustrates how to migrate storage groups with names that begin with ARU.

```sql
MIGRATE STOGROUPS
   WHERE NAME LIKE 'ARU%';
```

Example
The following example illustrates how to migrate all of the tables of the DEM database.

```sql
MIGRATE TABLES
   WHERE DBNAME = 'DEM';
```
Example

The following example illustrates how to migrate the tables that DEM created and include the parent databases and table spaces.

<table>
<thead>
<tr>
<th>MIGRATE TABLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCLUDE PARENTS</td>
</tr>
<tr>
<td>WHERE CREATOR = 'DEM';</td>
</tr>
</tbody>
</table>

REPLICATE WORKID statement

The REPLICATE WORKID statement copies work IDs and optionally applies specified changes to the objects. A task ID can contain only one REPLICATE WORKID statement. It can also contain one or more CHANGE statements.

Note

CHANGE MANAGER does not support importing the REPLICATE WORKID statement. For more information, see the chapter about importing DML statements in ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2.

Figure 223: REPLICATE WORKID statement

REPLICATE WORKID SET attribute 1 = value 1

Sets the attributes of a work ID to a value.

attribute 1

Specified the attribute that you want to be replicated. The attribute 1 variable can be one or more of the following keywords:
**REPLICATE WORKID statement**

- **WKOWNER**
- **WKNAME**

*value 1*

Specifies the new value for the attribute that you are changing.

**WHERE search condition 1**

Specifies the criteria that you use to define the scope for the replication. The search condition contains one or more of the following keywords:

- **WKOWNER**
- **WKNAME**
- **STATUS**
- **TYPE**
- **SECURE**

**CHANGE objectType**

Specifies the object type to be changed. The *objectType* variable can be one of the following keywords:

- **ALIASES**
- **CHECKS**
- **COLUMNS**
- **DATABASES**
- **INDEXES**
- **INDEXPARTS**
- **KEYCOLUMNS**
- **PARMS**
- **RELKEY**
- **RELS**
- **ROUTINES**
- **SEQUENCES**
- **STOGROUPS**
- **SYNONYMS**
- **TABLEPARTS**
- **TABLES**
- **TABLESPACES**
- **VIEWS**
Note
The keywords that are listed are created by the CM/PILOT component when it builds a DML statement. When modifying a DML statement, you can use the keywords as they are shown, singular forms of the keywords, or the actual DB2 catalog names.

**SET attribute = value**

Sets an attribute of an object to a value.

*attribute*

Specifies the attribute that you want to change. The term attribute as used in CM/PILOT DML is synonymous with the column names of DB2 catalog tables. In a REPLICATE WORKID statement, attribute refers to the values in the CHANGE MANAGER CD table columns. (For more information about the CD tables, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 1*.)

Table 268 on page 575 through Table 283 on page 590 list all of the CD table column attributes that can be modified using the CHANGE statement.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>AOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDNAMES</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNERS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGESLEVEL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CREATEDBY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ALNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_AOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_REM</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TBLOCK</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBLOCK</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBNAMES</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TOWNER</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
### Table 269: Check constraint CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CKNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_CKNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBOWNER</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

### Table 270: Column (table and view) CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVGLLEN</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CLNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COLL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>COLNO</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>COLTYPE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>FPROC</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_CLNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COLLL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COLNO</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COLSSCALE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COLTYPE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DEFAULTVALUE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DEFTI</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DTNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DTSHEMA</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_FPARM</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_FPROC</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Attribute</td>
<td>SET clause</td>
<td>WHERE clause</td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>NEW_LABEL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_NAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_NULLI</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PRIKEYNO</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_REMS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SUBTYPE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PRIKEYNO</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SEQUENCE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SUBTYPE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>TBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBNEW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>TBOwner</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TYPE</td>
<td>NA</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 271: Database CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPOOL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CREATEDBY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DBOwner</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_BPOOL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_CCSID</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DBOwner</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SGNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_IBPOOL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ROSHARE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ROSHARE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SGNAME</td>
<td>NA</td>
<td>X</td>
</tr>
</tbody>
</table>
### Table 272: Index CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMCSTATS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>BPOOL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANNEWLVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CLUST</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>COMPRESS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CREATEDBY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>FKEYCARD</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>INDEXTYPE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>IXNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IXOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_BPOOL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_CLUST</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COPY</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_CRULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ERULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_INDEXTYPE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_IXNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_IXOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PADDED</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PARTS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PGSIZE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PIECESIZE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_URULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_REMS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PARTS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>PGSIZE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>PRIMARY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>TBNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>Attribute</td>
<td>SET clause</td>
<td>WHERE clause</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>TBOWNER</td>
<td>NA</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 273: Index key CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CLNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COLNO</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>IXNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IXNEW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>IXOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_CLNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COLNO</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ORDERING</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ORDERING</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SEQUENCE</td>
<td>NA</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 274: Parm CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>LENGTH</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>LOCATOR</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_LENGTH</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_LOCATOR</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ORDINAL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PARMNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ROWTYPE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SCALE</td>
<td>NA</td>
<td>X</td>
</tr>
</tbody>
</table>
### Table 275: Relation CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Set clause</th>
<th>Where clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DELETERULE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>ENFORCE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>FKNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IXOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IXNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_FKNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DELETERULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_REFTBNNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_REFTBOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REFTBNNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REFTBOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Attribute</td>
<td>SET clause</td>
<td>WHERE clause</td>
</tr>
<tr>
<td>----------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>TBOWNER</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 276: Relation key CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>COLNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COLSEQ</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>FKNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>FKNO</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COLNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COLSEQ</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>REFTBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REFTBOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RLNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SEQUENCE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>TBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBNEW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>TBOWNER</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 277: Routine CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;LANGUAGE&quot;</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>&quot;NAME&quot;</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>&quot;TEXT&quot;</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>ACTIVE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>ADD_VERSION</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>ASUTIME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CDNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>Attribute</td>
<td>SET clause</td>
<td>WHERE clause</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>COLLID</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>COMMIT_ON_RETURN</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CONCUR_ACC_RES</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DATE_FORMAT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DBINFO</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DEBUG_MODE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DEC31</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DEC31_SCALE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DEFERPREP</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DEFERPREPARE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DEGREE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DETERMINISTIC</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DYNAMICRULES</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>ENCODING_CCSID</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>EXPLAIN</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>EXTERNAL_NAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>EXTERNAL_SECURITY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>FOR_UPDATE_CLAUSE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>IMMEDWRITE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>ISOLATION</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>JAR_ID</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>JARSCHEMA</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>JAVA_SIGNATURE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>KEEPDYNAMIC</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>MAX_FAILURE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>MIGAU</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ACTIVE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ASUTIME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COLLID</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COMMIT_ON_RETURN</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_CONCUR_ACC_RES</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DATE_FORMAT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>Attribute</td>
<td>SET clause</td>
<td>WHERE clause</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>NEW_DBINFO</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DEBUG_MODE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DEC31</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DEC31_SCALE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DEFERPREP</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DEFERPREPARE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DEGREE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DETERMINISTIC</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DYNAMICRULES</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ENCODING_CCSID</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_EXPLAIN</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_EXTERNAL_NAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_EXTERNAL_SECURITY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_FOR_UPDATE_CLAUSE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_IMMEDWRITE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ISOLATION</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_JAR_ID</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_JARSCHEMA</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_JAVA_SIGNATURE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_KEEPDYNAMIC</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_LANGUAGE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_MAX_FAILURE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_NAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_OPTHINT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ORIGIN</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_OWNER</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PACKAGEPATH</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PARAMETER_CCSID</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PARAMETER_STYLE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PARAMETER_VARCHARFORM</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PATHSCHEMAS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PROGRAM_TYPE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>Attribute</td>
<td>SET clause</td>
<td>WHERE clause</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>NEW_QUALIFIER</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_RELEASE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_REMS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_REOPT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_RESULT_SETS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ROUNDING</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_RUNOPTS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SCHEMA</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SPECIAL_REGS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SQL_DATA_ACCESS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_STAYRESIDENT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TIME_FORMAT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VALIDATE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VERSION</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_WLM_ENV_FOR_NESTED</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_WLM_ENVIRONMENT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEWTEXT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>OPTHINT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>ORIGIN</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>OWNER</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>PACKAGEPATH</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>PARAMETER_CCSID</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>PARAMETER_STYLE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>PARAMETER_VARCHARFORM</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>PATHSCHEMAS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>PROGRAM_TYPE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>QUALIFIER</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>RELEASE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>REMS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>REOPT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>RESULT_SETS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>ROUNCING</td>
<td>NA</td>
<td>X</td>
</tr>
</tbody>
</table>
Table 278: Storage group CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CREATEDBY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SGNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SGOWNER</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SGNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SGOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VCATNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VCATPASS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VCATNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>VCATPASS</td>
<td>NA</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 278: Storage group CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUNOPTS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SCHEMA</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SPECIAL_REGS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SPNEW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SQL_DATA_ACCESS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>STAYRESIDENT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SUBEC</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SUBET</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>TEXT_ROWID</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>TIME_FORMAT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>TYPE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>VALIDATE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>VERSION</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>WLM_ENV_FOR_NESTED</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>WLM_ENVIRONMENT</td>
<td>NA</td>
<td>X</td>
</tr>
</tbody>
</table>
Table 279: Synonym CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CREATEDBY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SYNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SYOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TBNNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TBOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SYNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SYOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBNNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBOWNER</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 280: Table CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARCHIVING_SCHEMA</td>
<td>X</td>
<td>NA</td>
</tr>
<tr>
<td>ARCHIVING_TABLE</td>
<td>X</td>
<td>NA</td>
</tr>
<tr>
<td>AUDITING</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CARD</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CREATEDBY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DATACAPTURE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DBNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DROPRESTRICT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>EPROC</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ARCHIVING_SCHEMA</td>
<td>X</td>
<td>NA</td>
</tr>
<tr>
<td>NEW_ARCHIVING_TABLE</td>
<td>X</td>
<td>NA</td>
</tr>
<tr>
<td>NEW_AUDITING</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Attribute</td>
<td>SET clause</td>
<td>WHERE clause</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>NEW_CCSID</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DATACAPTURE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DROPRESTRICT</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_EPROC</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_LABEL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_OBID</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_REF_TBNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_REF_TBOWNER</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_REMS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TBOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TSNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VERSIONING_NAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VERSIONING_SCHEMA</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VPROC</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>OBID</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>REF TBNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>REF TBOWNER</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>ROWAVG</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ROWAVGSTATS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ROWMAX</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>TBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBTYPE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TSNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>VERSIONING_NAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>VERSIONING_SCHEMA</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>VPROC</td>
<td>NA</td>
<td>X</td>
</tr>
</tbody>
</table>
Table 281: Table space CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMCSTATS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>BPOOL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CREATEDBY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IMPLICIT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>LOCKMAX</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>MAXPARTS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_BPOOL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_CCSID</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_CRULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DSSIZE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ERULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_IMPLICIT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_LOCKMAX</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_LRULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_MAXROWS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_MAXPARTS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_MEMMCLUST</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PARTS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SEGSIZE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TSNNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TSOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_TSTYPE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PARTS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SEGSIZE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>TSNNAME</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Table 282: Table space and index partition CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSOWNER</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>ALOCUNIT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>AVGLENKEY</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CARD</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CARDF</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>COMPRESS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DATACLASS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DEVTYPE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>DSVCAT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>FREEPAGE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>GBPCACHE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>IXNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IXNEW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>IXOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MGMTCLASS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>MOREVOLS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_ALOCUNIT</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COMPRESS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DATACLASS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_DEVTYPE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_FREEPAGE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_GBPCACHE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_MGMTCLASS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_MOREVOLS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PARTID</td>
<td>NA</td>
<td>X</td>
</tr>
</tbody>
</table>
### Table 283: View CD table column attributes used with CHANGE

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW_PCTFREE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PQTY</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SGNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_SQTY</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_STORCLASS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_STYPE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VCATNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VCATPASS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VOLORDER</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>PARTID</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PARTTYPE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>PCTCOMPRESSED</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PCTFREE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>PQTY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>ROWSPKEY</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ROWSPKEYF</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SEQUENCE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SGNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SGNEW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>SPACE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SQTY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>STORCLASS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>STYPE</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>TSNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TSNEW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>VCATNAME</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>VCATPASS</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>VOLORDER</td>
<td>NA</td>
<td>X</td>
</tr>
</tbody>
</table>

**Table 283: View CD table column attributes used with CHANGE**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTHIDSW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>Attribute</td>
<td>SET clause</td>
<td>WHERE clause</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>CDNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CDOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHANGELVL</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CHECK</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>COLSNTXT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>CREATEDBY</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_AUTHIDSW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_CHECK</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_COLSNTXT</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_LABEL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_PATHSCHEMAS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_QUALVIEW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>NEW_REMS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VWNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEW_VWOWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>QUALVIEW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>VWNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VWNEW</td>
<td>NA</td>
<td>X</td>
</tr>
<tr>
<td>VWOWNER</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

**value** 2

Specifies the new value for the attribute that you are changing. The value can be a constant or an expression. If you need more space than an input field provides on a panel, edit the DML statement. For example, you can change a NAME attribute with either of the following values: 'ABCD' or 'PROD'CONCATSUBSTR(NAME,5). You can change a numeric attribute such as PQTY with any of the following values: 123, PQTY*2, or SQTY *4.

**WHERE searchcondition** 2

Specifies the criteria that you use to define the scope for the change. The search condition consists of one or more predicates. You can use any search condition that is valid in a WHERE clause of an SQL statement. For information, see the IBM documentation.
Examples

Example
The following example illustrates how to replicate a work ID to another work ID. Changes made to the original work ID are saved as entries in the CHANGE MANAGER CD tables. When a work ID is replicated, all of the CD table entries are copied to the new work ID, with the exception of the creation date and time.

```
REPLICATE WORKID
SET WKOWNER='RIHJCB'
   WHERE WKOWNER='JCB'
   AND WKNAME='BWSDB001';
```

Example
The following example illustrates how to replicate a work ID and apply changes. The database in the original work ID (AALURU34) changes to another database name (AAMX1001) in the new work ID. This type of change is useful when you are making global changes in which the naming conventions need to be preserved.

```
REPLICATE WORKID
SET WKOWNER='RIHJCB'
   WHERE WKOWNER='JCB'
   AND WKNAME='BWSDB001';
CHANGE DATABASES
SET DBNAME='AAMX1001'
   WHERE DBNAME='AALURU34';
```

SET OMITUNICODE statement

The SET OMITUNICODE statement omits objects from the results table of a DML statement that have unsupported Unicode attributes.

The SET OMITUNICODE statement can appear anywhere in the DML_STRUCTURE_CHG (UPDATE statement) or DML_MIGRATE (LIKE or MIGRATE statement) script and can apply to all of the DML statements in the script. A task ID can contain only one SET OMITUNICODE statement.

Note
CHANGE MANAGER does not support importing the SET OMITUNICODE statement. For more information, see the chapter about importing DML statements in *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

Figure 224: SET OMITUNICODE statement

```
SET OMITUNICODE
```

592  *ALTER and CHANGE MANAGER for DB2 Reference Manual*
Example

Figure 225 on page 593 illustrates the use of the SET OMITUNICODE statement. In this example, if any of the tables with a creator of CRJ contains a Unicode attribute, CM/PILOT omits the table from the worklist and issues an informational message.

Figure 225: DML for omitting objects with long names

```
SET OMITUNICODE;
MIGRATE TABLES
  WHERE CREATOR LIKE 'CRJ%';
```

If the SET OMITUNICODE statement were not included in the DML for this example, CM/PILOT would stop processing the CM/PILOT worklist. To resume worklist processing, you would need to exclude the table from the WHERE clause of the DML statement or add the SET OMITUNICODE statement to the worklist.

Tip

You can copy the DML statement to omit objects from member ACMDMLF1 in the HLQ.BMCCNTL data set.

SET PRESERVELIMITKEY statement

The SET PRESERVELIMITKEY statement preserves the limit key values for the last partition when you convert index-controlled partitioned tables to table-controlled partitioned tables.

Figure 226: SET PRESERVELIMITKEY statement

```
SET PRESERVELIMITKEY ;
```

The SET PRESERVELIMITKEY statement can appear anywhere in the DML_STRUCTURE_CHG (UPDATE statement) script and can apply to all of the DML statements in the script.

A task ID can contain only one SET PRESERVELIMITKEY statement.

Example

The following example illustrates how to convert several tables from index-controlled partitioning to table-controlled partitioning by using the TCPART parameter. The SET PRESERVELIMITKEY statement preserves the limit key values for the last partition of each table.

```
UPDATE TABLES
SET PRESERVELIMITKEY;
SET TCPART = 'Y'
  WHERE CREATOR = 'CRJICP';
```
SET SPACE ESTIMATION statement

The SET SPACE ESTIMATION statement estimates the space for all table spaces and indexes in a database.

The statement can appear anywhere in the DML_STRUCTURE_CHG (UPDATE statement) or DML_MIGRATE (LIKE or MIGRATE statement) script and can apply to all of the DML statements in the script. A task ID can contain only one SET SPACE ESTIMATION statement. You can import the SET SPACE ESTIMATION statement in a DML file. For information, see the ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2.

Figure 227: SET SPACE ESTIMATION statement

SET SPACE ESTIMATION

Figure 227: SET SPACE ESTIMATION statement

SET SPACE ESTIMATION EXPLICIT or IMPLICIT

Specifies how CHANGE MANAGER estimates the space for objects in the DML statements.

EXPLICIT

Specifies that CHANGE MANAGER estimates the space for only the table spaces and indexes that are specified in DML statements for the TABLEPARTS and INDEXPARTS keywords (for example, UPDATE TABLEPARTS, LIKE INDEXPARTS, or MIGRATE TABLEPARTS). EXPLICIT is the default.

IMPLICIT

Specifies that CHANGE MANAGER estimates the space for all of the table spaces and indexes that are specified in the DML statements for DATABASES, TABLESPACES, TABLEPARTS, TABLES, INDEXPARTS, and...
INDEX keywords (for example, MIGRATE DATABASES or UPDATE TABLESPACES).

**PQTY = BMCSPACE or expression**

Specifies the value for the primary quantity.

**BMCSPACE**

BMCSPACE represents the value that is returned from CHANGE MANAGER space estimation for each table partition or index partition in the scope of the DML. BMCSPACE is the default.

**expression**

The expression must include the BMCSPACE keyword. For example, you can set the secondary quantity to be one fourth of the value that is returned from space estimation by specifying SQTY=BMCSPACE/4.

**SQTY = BMCSPACE or expression**

Specifies the value for the secondary quantity.

**BMCSPACE**

Represents the value that is returned from CHANGE MANAGER space estimation for each table partition or index partition in the scope of the DML. BMCSPACE is the default.

**expression**

The expression must include the BMCSPACE keyword. For example, you can set the secondary quantity to be one fourth of the value that is returned from space estimation by specifying SQTY=BMCSPACE/4.

For more information about space estimation, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

The following examples illustrate the use of the SET SPACE ESTIMATION statement.
Estimating space

*Example*

The following example illustrates how to estimate the space and to specify a value for the primary quantity. In this example, the product updates the primary and secondary quantities for all table spaces and indexes that are contained in databases that are named like "PSSOFT." The value of the primary quantity is set to three times the size of the space that is estimated. The value of the secondary quantity is set to one fourth of the size of the space that is estimated.

```
SET SPACE ESTIMATION IMPLICIT;
UPDATE TABLEPARTS
  SET PQTY = BMCSPACE * 3
  , SQTY = BMCSPACE/4
WHERE DBNAME LIKE 'PSSOFT%';
MIGRATE DATABASES
  WHERE NAME LIKE 'PSSOFT%';
```

Overriding the estimated space

*Example*

The following example illustrates how to estimate the space for all of the keywords in the script. In this example, the product estimates the primary and secondary quantities for table spaces that are contained in the "TEST" database. Because the INDEXPARTS keyword is specified in the UPDATE statement, the space is estimated explicitly. As a result, the primary quantity for the indexes in the "TEST" database is set to two times the value that is returned by space estimation. The secondary quantity for the indexes is set to one fourth of the value that is returned by space estimation.

```
SET SPACE ESTIMATION IMPLICIT;
UPDATE DATABASES
  SET BPPOOL = BP2
  WHERE NAME = 'TEST';
UPDATE INDEXPARTS
  SET PQTY = BMCSPACE * 2,
      SQTY = BMCSPACE/4
  WHERE DBNAME = 'TEST';
```
Implicitly estimating space

Example
The following example illustrates how to estimate the space implicitly for all of the keywords in the script. In this example, the product estimates the primary and secondary quantities for table spaces and indexes that are contained in the "DEMO" and "TEST" databases.

```
SET SPACE ESTIMATION IMPLICIT:
UPDATE DATABASES
SET BPOOL = BP2
    WHERE NAME = 'DEMO';

UPDATE TABLEPARTS
SET PCTFREE = 25
    WHERE DBNAME = 'TEST';
```

By default, the primary quantity is set to the size of the space that is estimated. The secondary quantity is set to one tenth of the size of the space that is estimated.

Explicitly estimating space

Example
The following example illustrates how to estimate the space explicitly. In this example, the product estimates the primary and secondary quantities only for table spaces and indexes that are contained in the "TEST" database.

```
SET SPACE ESTIMATION EXPLICIT:
UPDATE DATABASES
SET BPOOL = BP2
    WHERE NAME = 'DEMO';

UPDATE TABLEPARTS
SET PCTFREE = 25
    WHERE DBNAME = 'TEST';
```

The space is not estimated for table spaces and indexes that are contained in the "DEMO" database because the DATABASES keyword is specified in the UPDATE statement. When the EXPLICIT parameter is used in the SET SPACE ESTIMATION statement, the product estimates the space only for the table spaces and indexes that are specified in DML statements for the TABLEPARTS and INDEXPARTS keywords.

UPDATE statement

The UPDATE statement changes data structures.
You can import the UPDATE statement in a DML file. For information, see the *ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2*.

**Figure 228: UPDATE statement**

```sql
UPDATE objectType
  SET attribute = value
WHERE searchCondition
```

**UPDATE objectType**

Specifies the object type to be updated. The `objectType` variable can be one of the following keywords:

- ALIASES
- CHECKS
- COLUMNS
- DATABASES
- FIELDS
- FOREIGNKEYS
- INDEXES
- INDEXPARTS
- KEYCOLUMNS
- PARMS
- RELS
- ROUTINES
- SEQUENCES
- STOGROUPS
- SYNONYMS
- TABLEPARTS
- TABLES
- TABLESPACES
- TRIGGERS
- VIEWS
- VOLUMES
Note
The keywords that are listed are created by the CM/PILOT component when it builds a DML statement. When modifying a DML statement, you can use the keywords as they are shown, singular forms of the keywords, or the actual DB2 catalog names.

SET attribute = value

For details of the SET clause, see “SET clause” on page 560.

WHERE searchCondition

Specifies the criteria that you use to define the scope for the action.

For details of the WHERE clause, see “WHERE clause” on page 561.

Examples

The following examples illustrate the use of the UPDATE statement. For additional examples and tasks, see the ALTER and CHANGE MANAGER for DB2 User Guide, Volume 2.

Excluding NULL rows from indexes

Example
The following example illustrates how to exclude null rows from non-unique indexes.

```
UPDATE INDEXES
SET SPARSE = 'X'
WHERE IX_EXTENSION_TYPE IN ('V','S');
```

Updating check constraints

Example
The following example illustrates how to update a check constraint.

```
UPDATE CHECKS
SET CHECKCONDITION='(FO > 1.1E+12999999999)'
WHERE CHECKNAME='FO' AND
TBNAME='TBN3B' AND
TBOWNER='ALU076';
```
Updating columns

**Example**
The following example illustrates how to change the length of table and view columns that are named ZIPCODE and that are 5 to 9 characters in length.

```sql
UPDATE COLUMNS
SET LENGTH = 9
WHERE NAME = 'ZIPCODE' AND LENGTH = 5;
```

Updating databases

**Example**
The following example illustrates how to change the storage group of databases that currently use the OLDGROUP storage group and that have at least one table space that has over 100 active pages. The new storage group for the databases is named NEWGROUP.

```sql
UPDATE DATABASES
SET STGROUP = 'NEWGROUP'
WHERE STGROUP = 'OLDGROUP' AND
EXISTS (SELECT * FROM SYSIBM.SYSTABLESPACE
WHERE DBNAME = SYSIBM.SYSDATABASE.NAME AND
NACTIVE > 100);
```

Updating space allocation based on statistics

**Example**
The following example illustrates how to change the primary quantity for the partitions of a table space in which the amount of space that rows of data from active tables occupy is greater than 50 percent.

```sql
UPDATE TABLEPARTS
SET PQTY = PQTY * 2
WHERE PERCACTIVE > 50;
```

Updating synonyms

**Example**
The following example illustrates how to change the names of synonyms. In this example, the prefix of TEST is changed to PROD.

```sql
UPDATE SYNONYMS
SET NAME = 'PROD' CONCAT SUBSTR(NAME,5)
WHERE NAME LIKE 'TEST%';
```
Updating tables

Example
The following example illustrates how to change the creator of all of the tables in a database.

```
UPDATE TABLES
SET CREATOR = 'NEWCREATOR'
WHERE DBNAME = 'ABCD';
```

Example
The following example illustrates how to change the names of tables that begin with WMS to names that begin with MJF.

```
UPDATE TABLES
SET NAME = 'MJF' CONCAT SUBSTR(NAME,4)
WHERE NAME LIKE 'WMS%';
```

Changing explicitly created tables to implicitly created tables

Example
The following example illustrates how to move tables from explicitly created databases to implicitly created databases. The product performs the following actions:

- Drops the tables, and creates the tables in implicitly created table spaces and implicitly created databases
- Changes the names of the table space and database to <DEFLT>
- Changes partitioned indexes to non-partitioned indexes

```
UPDATE TABLES
SET DBNAME = '<DEFLT>'
WHERE DBNAME LIKE 'CRJICP%';
```

Converting tables from index-controlled partitioning to table-controlled partitioning

Example
The following example illustrates how to convert several tables from index-controlled partitioning to table-controlled partitioning by using the TCPART parameter. The TCPART parameter can also be used in the LIKE TABLES and MIGRATE TABLES statements.

```
UPDATE TABLES
SET TCPART = 'Y'
WHERE CREATOR = 'CRJICP';
```
Performance enhancements with the JOIN keyword

The CM/PILOT component supports the use of the JOIN keyword, which can significantly reduce the execution time of a DML statement that searches multiple tables.

The following rules govern the use of the JOIN keyword:

- The JOIN keyword must follow the object type in the DML statement and must precede a SET, INCLUDE, or WHERE clause.

- You can use either the actual DB2 catalog names or the CM/PILOT DML object-type keywords with the JOIN keyword and in the WHERE clause.

- You must use name prefixes if ambiguity exists in the DML statement. In the preceding example, name prefixes are required because the DBNAME attribute is common to both of the catalog tables that are used in the WHERE clause.

- You can use the JOIN keyword only to improve the performance of the WHERE clause. You cannot update any attribute of the specific catalog table following the JOIN keyword. For example, an error will occur in the example shown in Figure 229 on page 602 because the SQTY attribute is not an attribute of TABLESPACES (SYSIBM.SYSTABLESPACE).

Figure 229: Incorrect use of the JOIN keyword

```
MIGRATE TABLESPACES
JOIN TABLEPARTS
SET CREATOR = 'ABCD',
   SQTY = 123
WHERE TABLEPARTS.DBNAME = TABLESPACES.DBNAME AND
   TSNAME = NAME AND
   STORNAME = 'ABCDEFG';
```
DML for migrating table spaces

**Example**

The following DML statement migrates table spaces whose partitions use a specific storage group.

```
MIGRATE TABLESPACES INCLUDE ALL
   WHERE EXISTS (SELECT * FROM SYSIBM.SYSTABLEPART
      WHERE SYSIBM.SYSTABLEPART.DBNAME =
      SYSIBM.SYSTABLESPACE.DBNAME AND
      NAME = TSNAME AND
      STORNAME = 'ABCDEFG');
```

DML for migrating table spaces with a JOIN

**Example**

The following DML statement modifies the DML statement with a JOIN keyword, which should significantly reduce execution time.

This example uses CM/PILOT DML object-type keywords for the actual catalog table names.

```
MIGRATE TABLESPACES
   JOIN TABLEPARTS
      INCLUDE ALL
      WHERE TABLEPARTS.DBNAME = TABLESPACES.DBNAME AND
      TSNAME = NAME AND
      STORNAME = 'ABCDEFG';
```

DML for replicating changes in multiple databases

**Example**

The following statement uses a JOIN statement 3 to create CD entries for a new work ID. Both the SYSIBM.SYSDATABASE table and a LIKE statement with a wildcard character (%) are used to select all of the databases with names that are prefixed with AAMX1.

```
REPLICATE WORKID
   SET WKOWNER='RIHJCB'
      WHERE WKOWNER='JCB'
      AND WKNAME='BWSDB001';
CHANGE DATABASES
   JOIN SYSIBM.SYSDATABASE
      SET DBNAME=SYSIBM.SYSDATABASE.NAME
      WHERE DBNAME='AALURU34' AND
      SYSIBM.SYSDATABASE.NAME LIKE 'AAMX1%';
```
**DML for replicating work IDs to multiple databases**

**Example**

The following statement is similar to the preceding example, except that it illustrates the creation of a new work ID for each database that is named like AAMX1. The name for each of the new work IDs is the name of the original work ID suffixed with the name of the database.

To connect the changed database CD table entry with the replicated work ID, you must specify, in the SET statement, the association between the CD entry names or owners and the work ID names and owners.

```sql
REPLICATE WORKID
JOIN SYSIBM.SYSDATABASE
SET WKOWNER='RIHJCB',
    WKNAME='BWSDB001' CONCAT SUBSTR(SYSIBM.SYSDATABASE.NAME,1,8)
WHERE WKOWNER='JCB'
    AND WKNAME='BWSDB001' AND
    SYSIBM.SYSDATABASE.NAME LIKE 'AAMX1%';
CHANGE DATABASES
SET DBNAME=SYSIBM.SYSDATABASE.NAME
    CDNAME='BWSDB001' CONCAT SUBSTR(SYSIBM.SYSDATABASE.NAME,1,8)
WHERE DBNAME='AALURU34' AND
    SYSIBM.SYSDATABASE.NAME LIKE 'AAMX1%';
```

**Attributes used in SET and WHERE clauses**

The attributes that are shown in the CM/PILOT DML panels for SET and WHERE clauses have the same names and characteristics as the column names for the DB2 catalog tables (which are documented in the IBM documentation).

**Note**

You can use any column name of any catalog table in a WHERE clause. However, for SET clauses, you can use only the column names of the catalog tables that are listed in the following tables.

**Table 284: Alias attributes for SET and WHERE clauses**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LABEL</td>
<td>X</td>
<td>N/A</td>
</tr>
<tr>
<td>LOCATION</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REMARKS</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>N/A</td>
</tr>
</tbody>
</table>

<sup>a</sup> REMARKS is an alias for the attributes that are set on the CD table entry.
No input field exists in the panels for the REMARKS attribute. You must edit the SET clause to change it. The REMARKS column is updated by the COMMENT ON SQL statement.

Table 285: Check constraint attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHECKNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHECKCONDITION</td>
<td>X</td>
<td>N/A</td>
</tr>
<tr>
<td>CREATOR</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>TBNAME</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>TBOWNER</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
</tbody>
</table>

<sup>a</sup> You can use this attribute only in the SET clause for a LIKE statement.

Table 286: Column (table and view) attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLCARD</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COLCARDF</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COLNO</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COLTYPE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DEFAULTVALUE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>FLDPROC</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>FOREIGNKEY</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>HIDDEN&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>KEYSEQ</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LABEL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LENGTH</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LENGTH2</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NULLS</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
### Table 287: Database attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORDER</td>
<td>X&lt;sup&gt;b&lt;/sup&gt;</td>
<td>X&lt;sup&gt;b&lt;/sup&gt;</td>
</tr>
<tr>
<td>PERIOD</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REMARKS</td>
<td>X&lt;sup&gt;c&lt;/sup&gt;</td>
<td>N/A</td>
</tr>
<tr>
<td>SCALE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBCREATOR</td>
<td>X&lt;sup&gt;d&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>TBCREATOR</td>
<td>X&lt;sup&gt;d&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>TYPENAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TYPESCHEMA</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

- **a** This attribute applies only to a table.
- **b** This attribute only applies to an identity column in a table. No input field exists in the panels for the ORDER attribute. You must edit the SET and WHERE clauses to change it.
- **c** No input field exists in the panels for the REMARKS attribute. You must edit the SET clause to change it. The REMARKS column is updated by the COMMENT ON SQL statement.
- **d** You can use this attribute only in the SET clause for a LIKE statement.

### Table 288: Field (LIKE and UPDATE) attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLNO</td>
<td>N/A</td>
<td>X</td>
</tr>
<tr>
<td>COLTYPE</td>
<td>N/A</td>
<td>X</td>
</tr>
<tr>
<td>FLDPROC</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LENGTH</td>
<td>N/A</td>
<td>X</td>
</tr>
</tbody>
</table>

606  *ALTER and CHANGE MANAGER for DB2 Reference Manual*
### Table 289: Foreign key attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COLNO</td>
<td>N/A</td>
<td>X</td>
</tr>
<tr>
<td>COLSEQ</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DELETERULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REFTBCREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REFTBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RELNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBNAME</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

*a* You can use this attribute only in the SET clause for a LIKE statement.

### Table 290: Index attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPOOL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CLOSERULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CLUSTERING</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COMPRESS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COPY</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DEFINE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ERASERULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>FULLKEYCARDF</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NAME</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

*a* You can use this attribute only in the SET clause for a LIKE statement.
### Attributes used in SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>PADDED</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td>PGSIZE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PIECESIZE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REMARKS</td>
<td>X&lt;sup&gt;b&lt;/sup&gt;</td>
<td>N/A</td>
</tr>
<tr>
<td>TBCREATOR</td>
<td>X&lt;sup&gt;c&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>TBNAME</td>
<td>X&lt;sup&gt;c&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>UNIQUERULE</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

- **a** No input field exists in the panels for the PADDED attribute. You must edit the SET and WHERE clauses to change it.
- **b** No input field exists in the panels for the REMARKS attribute. You must edit the SET clause to change it. The REMARKS column is updated by the COMMENT ON SQL statement.
- **c** You can use this attribute only in the SET clause for a LIKE statement.

### Table 291: Index key attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COLSEQ</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IXCREATOR</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>IXNAME</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>ORDERING</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PERIOD</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

- **a** You can use this attribute only in the SET clause for a LIKE statement.

### Table 292: Index partition attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVGKEYLEN</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CARD</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CARDF</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>FREEPAGE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>GBPCACHE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IXCREATOR</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
</tbody>
</table>

- **a** You can use this attribute only in the SET clause for a LIKE statement.
### Attributes used in SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>IXNAME</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>PARTITION</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>PCTFREE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PQTY</td>
<td>X&lt;sup&gt;b&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>ROWSPKEY</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ROWSPKEYF</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SECQTYI</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SPACE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SQTY</td>
<td>X&lt;sup&gt;b&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>STORNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VCATNAME</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
</tbody>
</table>

- <sup>a</sup> You can use this attribute only in the SET clause for a LIKE statement.
- <sup>b</sup> The value of PQTY or SQTY can be an expression that includes the BMCSpace keyword. The BMCSpace keyword represents the value that is returned from CHANGE MANAGER space estimation.

### Table 293: Parm attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCSID</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENCODING_SCHEME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LENGTH</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LOCATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ORDINAL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>OWNER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PARMNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ROWTYPE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SCALE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SCHEMA</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SUBTYPE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TYPENAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TYPESCHEMA</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
### Table 294: Relation attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>VERSION</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

### Table 295: Routine attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DELETERULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENFORCED</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IXNAME</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>IXOWNER</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>REFTBCREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REFTBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RELNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBNAME</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
</tbody>
</table>

<sup>a</sup> You can use this attribute only in the SET clause for a LIKE statement.

### Table 295: Routine attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>APPLCOMPAT</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ARCHIVESENSITIVE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ASUTIME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>BUSTIMESENSITIVE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COLLID</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COMMIT_ON_RETURN</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CONCUR_ACC_RES</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DATE_FORMAT</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DBINFO</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DEBUG_MODE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DEC31</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DEFERPREP</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DEFERPREPARE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Attribute</td>
<td>SET clause</td>
<td>WHERE clause</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>DEGREE</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>DETERMINISTIC</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>DYNAMICRULES</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>ENCODING_CCSID</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>EXPLAIN</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>EXTERNAL_NAME</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>EXTERNAL_SECURITY</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>FENCED</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>IMMEDWRITE</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>ISOLATION</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>JAR_ID</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>JARSCHEMA</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>JAVA_SIGNATURE</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>KEEPDYNAMIC</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>LANGUAGE</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>MAX_FAILURE</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>NAME</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>NULL_CALL</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>OPTHINT</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>ORIGIN</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>OWNER</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PACKAGEPATH</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PARAMETER_CCSID</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PARAMETER_STYLE</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PARAMETER_VARCHARFORM</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PATHSCHEMAS</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PROGRAM_TYPE</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>QUALIFIER</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>RELEASE</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>REMARKS</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>REOPTVAR</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>RESULT_SETS</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>
### Attributes used in SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUNDAY</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RUNOPTS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SCHEMA</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SPECIAL_REGS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SQL_DATA_ACCESS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SYSTIMESENSITIVE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>STAYRESIDENT</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TIME_FORMAT</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VALIDATE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VERSION</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>WLM_ENV_FOR_NESTED</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>WLM_ENVIRONMENT</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 296: Sequence attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CACHE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COLTYPE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CYCLE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>INCREMENT</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MAXVALUE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MINVALUE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ORDER</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PRECISION</td>
<td>N/A</td>
<td>X</td>
</tr>
<tr>
<td>REMARKS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SCHEMA</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>START</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TYPENAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TYPESCHEMA</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
### Table 297: Storage group attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VCATNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MGMTCLAS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

### Table 298: Synonym attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBCREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TBNAME</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

### Table 299: Table attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARCHIVING_SCHEMA</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ARCHIVING_TABLE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>AUDITING</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CARD</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CARDF</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHECKS</td>
<td>N/A</td>
<td>X</td>
</tr>
<tr>
<td>CLUSTERTYPE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DATACAPTURE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>EDPROC</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENABLE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENCODING_SCHEME&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LABEL</td>
<td>X</td>
<td>N/A</td>
</tr>
</tbody>
</table>
### Attributes used in SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAINTENANCE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>OBID</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REMARKS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SPLIT_ROWS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TCPART</td>
<td>X</td>
<td>N/A</td>
</tr>
<tr>
<td>TSNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TYPE</td>
<td>X&lt;sup&gt;c&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>VALPROC</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VERSIONING_SCHEMA</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VERSIONING_TABLE</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

- **a** This attribute applies only to global temporary tables. If you specify a value for the encoding scheme for a table, the product adds TYPE = ‘G’ to the WHERE clause.
- **b** The SPLIT_ROWS attribute corresponds to the VOLATILE parameter.
- **c** You can use this attribute only in the SET clause for a LIKE statement.

### Table 300: Table space attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPOOL</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CLOSERULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DBNAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DEFINE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DSSIZE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENCODING_SCHEMA</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ERASERULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LOCKMAX</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LOCKPART</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LOCKRULE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LOG</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MAXPARTITIONS</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
### Table 301: Table space partition attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXROWS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SEGSIZE</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TYPE</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

### Table 302: Trigger attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATEDBY</td>
<td>N/A</td>
<td>X</td>
</tr>
<tr>
<td>CREATEDTS</td>
<td>N/A</td>
<td>X</td>
</tr>
<tr>
<td>DBID</td>
<td>N/A</td>
<td>X</td>
</tr>
</tbody>
</table>
### Table 303: View attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHECK</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CREATOR</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LABEL</td>
<td>X</td>
<td>N/A</td>
</tr>
<tr>
<td>NAME</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REMARKS</td>
<td>X</td>
<td>N/A</td>
</tr>
</tbody>
</table>

* a No input field exists in the panels for the REMARKS attribute. You must edit the SET clause to change it. The REMARKS column is updated by the COMMENT ON SQL statement.

### Table 304: Volume attributes for SET and WHERE clauses

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SET clause</th>
<th>WHERE clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGC CREATOR</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>SG NAME</td>
<td>X&lt;sup&gt;a&lt;/sup&gt;</td>
<td>X</td>
</tr>
<tr>
<td>VOLID</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

* a You can use this attribute only in the SET clause for a LIKE statement.
-AMS worklist command 261
-ANLY CM/PILOT worklist command 352
-AUTH worklist command 262
-BASF worklist command 264
-BEGG worklist command 265, 309
-BEGP worklist command 265, 310
-BEGU worklist command 267, 310
-BIND worklist command 267
-BJCL CM/PILOT worklist command 353
-BMCC command
creating migration files 307
EXPORT command 307
-BMCC worklist command
DSNAME parameter 270
dynamically allocating data sets 270
OUTPUT descriptor 270
parameters for a partition-by-growth table space 272
processing in parallel 269
used in a parallel worklist 265
used with -SPAC worklist command 337
used with partition-level image copies 271
-BMCD worklist command
DSNAME parameter 274
dynamically allocating data sets 274
FORMAT BMCLOAD parameter 275
OUTPUT descriptor 274, 279
processing in parallel 273
unloading LOB data 279
unloading XML data 279
used in a parallel worklist 265
used with -UNLB worklist command 279, 287
-BMCK worklist command
processing in parallel 280
used in a parallel worklist 265
-BMCL worklist command
FORMAT BMCUNLOAD parameter 281
LOAD DATA REPLACE INTO TABLE syntax 289
LOAD DATA RESUME YES INTO TABLE REPLACE syntax 289
LOAD DATA RESUME YES INTO TABLE syntax 289
loading LOB data 287
loading XML data 287
migrating only data with FORCELOADREPLACE 289
processing in parallel 280
REMOVEDECIMAL parameter 292
used in a parallel worklist 265
used with -LOLB worklist command 279, 287
used with -SPAC worklist command 337
used with multiple SYSUT data sets 282
used with partition-level image copies 283
-BMCR worklist command
used with -SPAC worklist command 337
used with multiple SYSUT data sets 293
used with partition-level image copies 295
-BMCS worklist command
processing in parallel 297
used in a parallel worklist 265
-BMCU worklist command 357
-BMCV command
IMPORT command 302
migrating image copy data 302
-BMCV worklist command
DSNAME parameter 300
dynamically allocating data sets 300
OPTION parameter 301
OUTPUT descriptor 300
used in a parallel worklist 265
used with -SPAC worklist command 337
used with partition-level image copies 301
-CHEK worklist command
used in a parallel worklist 265
-CMD worklist command 307
-COMP CM/PILOT worklist command 353
-COPY worklist command
  DSN parameter 308
dynamically allocating data sets 308
parameters for a partition-by-growth table
  space 309
  TEMPLATE descriptor 308
  used with -SPAC worklist command 337
-DBUG worklist command 357
-DEST worklist command 375
-DML CM/PILOT worklist command 354
-DSN1 worklist command 358
-ENDG worklist command 265, 309
-ENDP worklist command 265, 310
-ENDU worklist command 267, 310
-ERR worklist command 310
-GLID worklist command 311
-GOTO worklist command 311
-IMP CM/PILOT worklist command 354
-ISMT worklist command 312
-ISMX worklist command 313
-JCLP worklist command 314
  COPYPARTS parameter 315
  IPPARTS parameter 317
  KEYLEN parameter 317
  NUMPARTS parameter 315
  specifying SYSUT data sets 316
  used with -BMCL worklist command 282, 316
  used with -BMCR worklist command 293, 316
  used with -UNRC worklist command 348
  used with partition-level image copies 315
-LCMD worklist command 318
-LDXT worklist command 319
-LOAD worklist command
  IGNOREFIELDS YES parameter 322
  LOAD DATA REPLACE INTO TABLE syntax 324
  LOAD DATA RESUME YES INTO TABLE
  syntax 324
  loading LOB data 322
  loading XML data 322
  migrating only data with
  FORCELOADREPLACE 324
  REUSE parameter 326
-LOLB worklist command
  used with -BMCL worklist command 279, 287
-MERG worklist command 359
-MIGR worklist command 327
-MODI worklist command 359
-NOOP worklist command 360
-OPTS CM/PILOT worklist command 355
-ORGN worklist command 375
-QUI worklist command 360
-RBLD worklist command 329
-REBD worklist command 329
-REOR worklist command 330
-REPL CM/PILOT worklist command 356
-REPO worklist command 361
-REPX worklist command 361
-REXC worklist command 332
-RNAM worklist command 333
-RNST worklist command
  IBM RUNSTATS utility 333
  parameters for a partition-by-growth table
  space 333
  used in a parallel worklist 265
-SETA worklist command 334
-SETP worklist command 335
-SETS worklist command 334, 336
-SPAC worklist command 337
-SPBX worklist command 338
-SQL DELETE command 321, 340
-SQL worklist command 340
-SQLM worklist command 361
-SQLP worklist command 341
-SSID migrate-type worklist command 342
-STOP worklist command 343
-STOS worklist command 362
-SYNC alter-type worklist command 343
-SYNC migrate-type worklist command 343
-TASK CM/PILOT worklist command 357
-TIME CM/PILOT worklist command
  file creation time 345, 375
-UNLB worklist command
  used with -BMCD worklist command 279, 287
-UNLI worklist command
  deleting data sets 345
dynamically allocating data sets 345
  MAXERR parameter 345
  TEMPLATE descriptor 345, 346
  unloading LOB data 346
  unloading XML data 346
  using multitasking 346
-UNRC worklist command
  DDNAME parameter 348
  DEFINENO parameter 348
  DSNAME parameter 348
dynamically allocating data sets 348
  EMPTY parameter 348
  establishing full-recovery baselines 348
  FILEREF parameter 348
-WKID (work ID) command
  alter or migrate worklist 350
"LANGUAGE" attribute
  modifying for a routine in a REPLICATE
  WORKID statement 581–585
"NAME" attribute
  modifying for a routine in a REPLICATE
  WORKID statement 581–585
"TEXT" attribute
  modifying for a routine in a REPLICATE
  WORKID statement 581–585
**PREFIX** TEMPLATE descriptor variable
  PREFIX symbolic variable 152
SYSUID symbolic variable 153
UID symbolic variable 154
USERID symbolic variable 155
ZPREFIX symbolic variable 157
ZSYSID symbolic variable 157
ZUSER symbolic variable 157
\xd2 NAME\xd3 attribute
  modifying for a parm column in a REPLICATE
  WORKID statement 579, 580
+1 OUTPUT descriptor variable 147
+1 TEMPLATE descriptor variable 147

2MEGSQL AEXIN keyword 55
2MEGSQL POF keyword 189

A
A (after) action code 32
A (analyze) action code 10 d 32
ACM AEXIN keyword 55
ACM_AMS POF keyword 139, 189
ACM_ANALYSIS_SYSTOUT POF keyword 189
ACM_BASDIAG POF keyword 139, 190
ACM_BRPRTDIAG POF keyword 140, 190
ACM_BRPRTDPSN POF keyword 140, 190
ACM_CDLDPSN POF keyword 140, 190
ACM_CDLPSS POF keyword 140, 190
ACM_CDLDU POF keyword 140, 190
ACM_CPLPDIAG POF keyword 140, 191
ACM_CPLCDCLO POF keyword 191
ACM_CPLPDIA POF keyword 141, 191
ACM_CPLWDSN POF keyword 141, 191
ACM_CPLWDSNDO POF keyword 191
ACM_DBRMI POF keyword 141, 191
ACM_DBRM2 POF keyword 141, 191
ACM_DBRM3 POF keyword 141, 191
ACM_DYNSORTW_NUM POF keyword 192
ACM_DYNSORTW_UNIT POF keyword 192
ACM_GLID POF keyword 142, 192
ACM_IBMR_MAP_REQ POF keyword 192
ACM_IMPDIAG POF keyword 142, 192
ACM_JDSN POF keyword 142, 192
ACM_JDSNB POF keyword 142, 193
ACM_JDSNDBG POF keyword 142, 193
ACM_JDSNBRD POF keyword 142, 193
ACM_JDSNC POF keyword 142, 193
ACM_JDSNCPL POF keyword 142, 193
ACM_JDSNCPO POF keyword 194
ACM_JDSNE POF keyword 142, 194
ACM_JDSNEI POF keyword 142, 194
ACM_PARALLEL_MAXINIT AJXPOFIN keyword 63
ACM_PARALLEL_MAXINIT=3 194
ACM_PARALLEL_MININIT AJXPOFIN keyword 63
ACM_PARALLEL_MININIT POF keyword 194
PARALLEL AEXPIN keyword 63
PARALLEL ALUIN keyword 119
ACM_PARALLEL_WORKLST POF keyword 195
ACM_PARALLEL_XIMGRP AJXPOFIN keyword 63
ACM_PARALLEL_XIMGRP POF keyword 195
ACM_PARALLEL_XIMPROC AJXPOFIN keyword 64
ACM_PARALLEL_XIMPROC POF keyword 195
ACM_PARALLEL_XIMSTRT AJXPOFIN keyword 64
ACM_PARALLEL_XIMSTRT POF keyword 195
ACM_PARALLEL_XIMTRCE AJXPOFIN keyword 63
ACM_PARALLEL_XIMTRCE POF keyword 195
ACM_PARALLEL_XIMP Sel AJXPOFIN keyword 64
ACM_PARALLEL_XIMP Sel POF keyword 195
ACM_PARALLEL_XIMSEL ATXPOFIN keyword 64
ACM_PARALLEL_XIMSEL ATXPOFIN keyword 195
ACM_PARALLEL_XIMSEL POF keyword 195
ACM_PARALLEL_XIMSTRT AJXPOFIN keyword 63
ACM_PARALLEL_XIMSTRT POF keyword 195
ACM_PARALLEL_XIMTRCE AJXPOFIN keyword 63
ACM_PARALLEL_XIMTRCE POF keyword 195
ACM_PIC POF keyword 143, 196
ACM_SDSN POF keyword 144, 196
ACM_SDSNE POF keyword 144, 196
ACM_WDSN POF keyword 145, 196
ACM_WLORDER POF keyword 110, 196
ACM_WLORDER MSG POF keyword 197
ACM_WLP S POF keyword 145, 197
ACM_WLSS POF keyword 145, 197
ACM_WLU POF keyword 145, 197
ACMDMLF1 member of HLQ.BMCCNTL data set 592
action codes, list of 32
ACTIVATE attribute 49
ACTIVE attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585

ACTWRK_DATACLASS 197
ACTWRK_MGMTCLASS 197
ACTWRK_PRIQTY 198
ACTWRK_STORCLASS 197
ACTWRK_UNIT 198
ADD_VERSION attribute
  modifying for a routine in a REPLICATE WORKID statement 581–585

ADDLOAD1 POF keyword 198
ADDLOAD2 POF keyword 198
AEXIN input stream 53
AEXIN keywords, list of 55
AEXPIN keywords, list of 62
AEXPRINT
  CM/PILOT deleted sync table entries report 563
  CM/PILOT deleted work ID report 565
AI command 26
AJX4PART SLIB variable 151
AJX4YDDD SLIB variable
  DATEJ symbolic variable 146
  JDATE symbolic variable 148
  JU symbolic variable 149
  JUL4Y symbolic variable 149
  YE symbolic variable 156
  YEAR symbolic variable 156
  YYYYDDD symbolic variable 157
AJX5PART SLIB variable 151
AJXBMCPP SLIB variable 146
AJXCR SLIB variable
  CR symbolic variable 146
  IXCR symbolic variable 148
AJXDB SLIB variable 146
AJXDB2V2 SLIB variable 146
AJXDB2V3 SLIB variable 147
AJXDDN SLIB variable 148
AJXDDOPT SLIB variable 147
AJXDDSQC SLIB variable
  DDSEQ symbolic variable 147
  SEQ symbolic variable 152
  SQ symbolic variable 153
AJXDSN44 SLIB variable 146
AJXFCMD SLIB variable 147
AJXFJCHR SLIB variable 148
AJXGDPGC SLIB variable 147
AJXHM SLIB variable 147

AJXHMS SLIB variable
  HMS symbolic variable 147
  HO symbol ic variable 147
  HOUR symbolic variable 147
  JHMS symbolic variable 148
  MI symbolic variable 151
  MINUTE symbolic variable 151
  SC symbolic variable 152
  SEC symbolic variable 152
  SECOND symbolic variable 152
  TI symbolic variable 153
  TIME symbolic variable 153
  TIME4 symbolic variable 153
AJXIC SLIB variable 148
AJXIN input stream 53
AJXIN keywords, list of 64
AJXIX SLIB variable 148
AJXIXNOD SLIB variable 148
AJXIXSPC SLIB variable 148
AJXJAID SLIB variable 146
AJXJCLEN Ajxin keyword 65
AJXJDDN SLIB variable 147
AJXJJDLD SLIB variable 148
AJXJOBNM SLIB variable 149
AJXJOBT SLIB variable 149
AJXJPCOD SLIB variable 149
AJXJOQID SLIB variable
  JQID symbolic variable 149
  WKID symbolic variable 156
  WORKID symbolic variable 156
AJXJSSID SLIB variable
  ATTACH symbolic variable 146
  GRPNM symbolic variable 147
  SS symbolic variable 153
AJXJYMD SLIB variable 150
AJXLLQ SLIB variable 150
AJXLR SLIB variable
  LOCREM symbolic variable 150
  LR symbolic variable 150
  TYPE symbolic variable 154
AJXMEMBER SLIB variable 150
AJXMEMBR SLIB variable 150
AJXMSSID SLIB variable 151
AJXOBJT SLIB variable 152
AJXOBNAME SLIB variable 151
AJXOBNOD SLIB variable 151
AJXOBT SLIB variable 151
AJXODS44 symbolic variable 146
AJXPARTC SLIB variable
DSNUM symbolic variable 147
LDSNUM symbolic variable 150
LPART symbolic variable 151
PA symbolic variable 151
PART symbolic variable 151
AJXPB SLIB variable 152
AJXPGMR SLIB variable 152
AJXPOFIN input stream 53
AJXRHLQ SLIB variable 152
AJXRUNTP SLIB variable 152
AJXSEQ# SLIB variable 152
AJXSPNAM SLIB variable
SN symbolic variable 152
SPNAME symbolic variable 153
TSIX symbolic variable 154
AJXSSID SLIB variable
JSSID symbolic variable 149
SSID symbolic variable 153
AJXSTEPC SLIB variable 153
AJXSTEPN SLIB variable 153
AJXSYSM SLIB variable 153
AJXTBCR SLIB variable 153
AJXTBCRE SLIB variable 153
AJXTBNAM SLIB variable 153
AJXTBNOD SLIB variable 153
AJXTS SLIB variable 154
AJXTSCR SLIB variable 154
AJXTSSID SLIB variable 154
AJXTU1 SLIB variable 154
AJXTU2 SLIB variable 154
AJXTU3 SLIB variable 154
AJXUCMD SLIB variable 154
AJXUDOPT SLIB variable 154
AJXULLQ SLIB variable 154
AJXUPART SLIB variable 155
AJXUTID SLIB variable 155
AJXUVR1 SLIB variable 155
AJXUVR2 SLIB variable 155
AJXUVR3 SLIB variable 155
AJXUVR4 SLIB variable 155
AJXUVR5 SLIB variable 155
AJXVCAT SLIB variable 155
AJXWKID SLIB variable
JOBNAME symbolic variable 149
WORKID8 symbolic variable 156
AJXWKOWN SLIB variable 156
AJXYMD SLIB variable
DA symbolic variable 146
DATE symbolic variable 146
DAY symbolic variable 146
DT symbolic variable 147
MO symbolic variable 151
MONTH symbolic variable 151
YMD symbolic variable 156
AJXYYDDD SLIB variable
DDD symbolic variable 147
JD symbolic variable 148
JDAY symbolic variable 148
JULIAN symbolic variable 150
YY symbolic variable 156
YYDDD symbolic variable 156
AL (aliases) action code 32
AL object type 19
aliases
attributes for SET and WHERE clauses 604, 605
DML keyword 559
modifying attributes in a REPLICATE
WORKID statement 573
performable actions 557
profile change rules 45
ALIASES DML keyword 559
aliasName variable 377
aliasOwner variable 377
ALID symbolic variable 146
ALLOC installation option 139, 161
ALLOWICP ALUIN keyword 131, 141, 144
See also STOPICP ALUIN keyword
ALLOWREBUILD ALUIN keyword
ALLOWREORG ALUIN keyword 73
ALLSTATSUPD ALUIN keyword 73, 145
ALNAME attribute 575
ALOCUNIT attribute 589, 590
ALOWNER attribute 575
ALTER ALIAS statement 379
ALTER AUXILIARY TABLE statement 380
ALTER CHECK statement 381
ALTER command 26
ALTER DATABASE statement 383
ALTER FOREIGN KEY statement 385
ALTER INDEX statement 388
ALTER PROCEDURE REGENERATE statement 122
ALTER PROCEDURE statement 398
ALTER SEQUENCE statement 423
ALTER STOGROUP statement 426
ALTER SYNONYM statement 427
ALTER TABLE statement 429
ALTER TABLESPACE statement 452
ALTER TRIGGER statement 462
ALTER UNIQUE CONSTRAINT statement 382
ALTER VIEW statement 466
ALTERADDVOL ALUIN keyword 73
ALTERID AEXIN keyword 55
ALU AEXIN keyword 55
ALUIN keywords, list of 71
AMS AJXIN keyword 65
AMS installation option 77, 139
AMSDELTEI ALUIN keyword 73, 99
Analysis CM/PILOT worklist command 352
ANALYZE AJXIN keyword 65
ANLBASLINE AJXIN keyword 65
ANLDIAG AJXIN keyword 65
ANLMIGPROFILE AJXIN keyword 65
ANP installation option 139, 162
APPLCOMPAT attribute for SET and WHERE clauses in a routine 610–612
ARCH_DATACLASS POF keyword 198
ARCH_DATACLASS_ALT POF keyword 198
ARCH_EXPDT 199
ARCH_MGMTCLASS POF keyword 199
ARCH_MGMTCLASS_ALT 199
ARCH_PREFIX POF keyword 199
ARCH_PRIQTY 199
ARCH_RETPD POF keyword 199
ARCH_SECQTY 199
ARCH_STACK POF keyword 199
ARCH_STORCLASS POF keyword 199
ARCH_STORCLASS_ALT POF keyword 200
ARCH_THRESH POF keyword 200
ARCH_UNIT POF keyword 200
ARCH_UNIT_ALT POF keyword 200
ARCHIVESENSITIVE attribute for SET and WHERE clauses in a routine 610–612
ARCHIVING_SCHEMA attribute 586, 613
ARCHIVING_TABLE attribute 586, 613
ARCHSTAK AJXIN keyword 65
ASU AEXIN keyword 56
ASU_LOGD_DATAC POF keyword 200
ASU_LOGD_LOGDSN POF keyword 201
ASU_LOGD_MGMTC POF keyword 200
ASU_LOGD_PRIQTY POF keyword 200
ASU_LOGD_SECQTY POF keyword 200
ASU_LOGD_STORC POF keyword 201
ASU_LOGD_UNIT POF keyword 201
ASU_UIMSRVHOST POF keyword 201
ASU_UIMSRVPORT POF keyword 201
ASU_XP_UIMSRVTIMEOUT POF keyword 201
ASUTIME attribute 49
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE WORKID statement 581–585
ATLOGPOINT option, EXPORT command 307
ATTACH OUTPUT descriptor variable
ATTACH symbolic variable 146
ZSYSID symbolic variable 157
ATTACH symbolic variable 146
ATTR (screen attributes) command 26
ATTR RESET (reset screen attributes) command 26
attributes for change rules, list of 49
attributes for DML SET and WHERE clauses, list of 604
AUC AEXIN keyword 56
AUDIT attribute 45, 49
AUDITING attribute 586, 587, 613, 614
AUTHIDSW attribute 590, 591
authorization switching
-AUTH worklist command 262
authorizationID variable 377
AUTHSW ALUIN keyword 74, 139
AUTHSW installation option 162
AUTHSW ALUIN keyword 74
AUTHSWGLID ALUIN keyword 74
AUTHSWOFF ALUIN keyword 74
NOAUTHSW ALUIN keyword 99
overriding keywords 139
AUTHSWGLID ALUIN keyword 74, 139
AUTHSWOFF ALUIN keyword 74, 139
AUX option, EXPORT command 271
AVGKEYLEN attribute
for SET and WHERE clauses in an index partition 608, 609
AVGLEN attribute 576, 577
AVGLENKEY attribute 589, 590
AX (auxiliary objects) action code 32
AX (auxiliary objects) command 26

B

B (before) action code 32
B (browse) action code 32
BASDIAG installation option 139
BASE installation option 139, 163
BASELINE-DELETE parameter, OVERRIDE ALUIN keyword 111
BASICUNLOAD ALUIN keyword 74
Batch Execution JCL Generation CM/PILOT
worklist command 353
begin global commands worklist command 265, 309
begin parallel processing worklist command 265, 310
begin unit of work worklist command 267, 310
begin global commands worklist command 265, 309
begin parallel processing worklist command 265, 310
bind worklist command 267
BIND AJXIN keyword 65
BINDFAIL AEXIN keyword 56
BINDFAIL POF keyword 201
BINDONIX 140, 163
BLNAME ALUIN keyword 75
BLPROFILE ALUIN keyword 75
BLRECOVER ALUIN keyword 75
BLRECOVERPOINT ALUIN keyword 75
BLRP_DATACLASS POF keyword 201
BLRP_DATACLASS_ALT POF keyword 202
BLRP_EXPDT POF keyword 202
BLRP_MGMTCLASS POF keyword 202
BLRP_MGMTCLASS_ALT POF keyword 202
BLRP_PREFIX POF keyword 202
BLRP_PRIQTY POF keyword 202
BLRP_RETPD POF keyword 202
BLRP_SECQTY POF keyword 202
BLRP_STACK POF keyword 202
BLRP_STORCLASS POF keyword 203
BLRP_STORCLASS_ALT POF keyword 203
BLRP_THRESH POF keyword 203
BLRP_UNIT POF keyword 203
BLRP_UNIT_ALT POF keyword 203
BLRP_TERMC installation option 133, 140, 164
BLWORKID ALUIN keyword 75
BMC Software utilities
  BASIC UNLOAD 268
  CHECK PLUS 280
  LOADPLUS 280
  NGT Copy 269
  NGT Recover 298
  REORG PLUS 293
  UNLOAD PLUS 273
BMC utilities, invoking
  DASD MANAGER PLUS 357
  BMC_CHECK_LOAD POF keyword 203
  BMC_CHECK_OPTS AJXPOFIN keyword
    CHECKDOPT AJXIN keyword 65
    CHECKOPT AEXIN keyword 57
  BMC_COPY_LOAD POF keyword 203
  BMC_COPY_OPTS AJXPOFIN keyword
  BMC_COPY_OPTS POF keyword 204
COPYDOPT AJXIN keyword 66
COPYOPT AEXIN keyword 57
BMC_COPY_OPTS POF keyword 204
BMC_LOAD_LOAD POF keyword 204
BMC_LOAD_OPTS AJXPOFIN keyword
  LOADDOPT AJXIN keyword 67
  LOADOPT AEXIN keyword 59
BMC_LOAD_OPTS POF keyword 204
BMC_RECOVER_OPTS POF keyword 204
BMC_RECOVERY_LOAD POF keyword 204
BMC_RECOVERY_OPTS AJXPOFIN keyword
  RECOVERDOPT AJXIN keyword 69
  RECOVEROPT AEXIN keyword 60
BMC_RECOVERY_OPTS POF keyword 204
BMC_REORG_LOAD POF keyword 205
BMC_REORG_OPTS AJXPOFIN keyword
  REORG AEXIN keyword 60
  REORGDOPT AJXIN keyword 69
BMC_REORG_OPTS POF keyword 204
BMC_REORG_XBMID POF keyword
BMC_UNLOAD_LOAD POF keyword 205
BMC_UNLOAD_OPTS 205
BMC_UNLOAD_OPTS AJXPOFIN keyword
  UNLOADDOPT AJXIN keyword 70
  UNLOADOPT AEXIN keyword 62
BMCCHECK ALUIN keyword 75, 140
BMCCHECK installation option 164
  BMCCHECK ALUIN keyword 75
  BMCCHECK ALUIN keyword 92
overriding keywords 140
BMCCOPY ALUIN keyword 75, 140
BMCCOPY installation option 92, 164
  BMCCOPY ALUIN keyword 75
  NOCOPY ALUIN keyword 100
  NOCOPYFORCE ALUIN keyword 100
overriding keywords 140
BMCCCP symbolic variable 146
BMCFASTL installation option 164
  BMCFASTLOAD ALUIN keyword 76
  NOBMCFASTLOAD ALUIN keyword 99
overriding keywords 140
BMCFASTLOAD ALUIN keyword 75, 140
BMCLoad ALUIN keyword 76, 140
BMCLoad installation option 164
  BMCLoad ALUIN keyword 76
  IBMLOAD ALUIN keyword 92
overriding keywords 140
BMCREBUILD ALUIN keyword 76, 143
BMCREORG ALUIN keyword 76, 143
BMCSPACE keyword 594
BMCSSTATS attribute 578, 579, 588, 589
BMCSTATS utility, -BMCS worklist command 297
BMCSTATS YES parameter 292, 297
BMCSTATSUPD ALUIN keyword 76, 145
BMCUNLD installation option 165
   BASICUNLOAD ALUIN keyword 74
   BMCUNLOAD ALUIN keyword 77
   IBMUNLOAD ALUIN keyword 92
   overriding keywords 140
BMCUNLOAD ALUIN keyword 77, 140
BPOOL attribute
   for SET and WHERE clauses in a database 606
   for SET and WHERE clauses in a table space 614, 615
   for SET and WHERE clauses in an index 607, 608
   modifying for a database in a REPLICATE WORKID statement 577
   modifying for a table space in a REPLICATE WORKID statement 588, 589
   modifying for an index in a REPLICATE WORKID statement 578, 579
BPOOLIX installation option 140, 165
BPOOLTS installation option 140, 165
BROWSE command 26
BRPTDIAG installation option 140
BRPTDSN installation option 140
bufferPoolName variable 377
BUFFPOOLIX attribute 45, 49
BUFFPOOLSTS attribute 45, 50
BUILDTEMPCOPY ALUIN keyword 77, 100
BUSTIMESENSITIVE attribute
   for SET and WHERE clauses in a routine 610–612
bypass worklist command 311

C

C (copy) action code 32
C (create) action code 32
CACHE attribute 612
CAL (create alias) action code 32
CAL (create alias) command 26
CANCEL (cancel action) command 27
CARD attribute
   for SET and WHERE clauses in a table 613, 614
   for SET and WHERE clauses in a table space partition 615
   for SET and WHERE clauses in an index partition 608, 609
   modifying for a index partition in a REPLICATE WORKID statement 589, 590
   modifying for a table in a REPLICATE WORKID statement 586, 587
   modifying for a table space partition in a REPLICATE WORKID statement 589, 590
   modifying for an index partition in a REPLICATE WORKID statement 589, 590
CAT_LOAD POF keyword 205
CATALOG ALUIN keyword 77
catalog indirection, used by CM/PILOT 557
catalogName variable 377
CATAUDIT AEXIN keyword 56, 140
CATAUDIT installation option 140, 166
CATDOPT AEXIN keyword 56
CATDOPT AJXIN keyword 65
CATRECOV installation option 57
CATRECOVER AEXIN keyword 57, 140
CATUTIL AEXIN keyword 57
CAX (create auxiliary object) action code 32
CAX (create auxiliary object) command 27
CC (block copy) action code 32
CCK (create check constraint) action code 32
CCK (create check constraint) command 27
CCSID attribute
   for SET and WHERE clauses in a parm 609, 610
CCSID installation option 140, 166
CD table column attributes
   used with REPLICATE WORKID statement 576, 577
CD table column attributes, used with REPLICATE WORKID statement 576–581, 585–591
CDB (create database) action code 32
CDB (create database) command 27
CDL (Change Definition Language)
delimiting reserved words 376
reserved words 376
syntax diagrams for statements 377
CDL commands
  -CDL 376
  -DEST 375
  -DEST (CDL destination subsystem) 375
  -ORGN 375
CDL origin subsystem worklist command 375
CDL statements
ALTER ALIAS 379
ALTER AUXILIARY TABLE 380
ALTER CHECK 381
ALTER DATABASE 383
ALTER FOREIGN KEY 385
ALTER INDEX 388
ALTER PROCEDURE 398
ALTER SEQUENCE 423
ALTER STOGROUP 426
ALTER SYNONYM 427
ALTER TABLE 429
ALTER TABLESPACE 452
ALTER TRIGGER 462
ALTER UNIQUE CONSTRAINT 382
ALTER VIEW 466
CREATE ALIAS 470
CREATE AUXILIARY TABLE 471
CREATE CHECK 472
CREATE DATABASE 474
CREATE FOREIGN KEY 477
CREATE INDEX 479
CREATE PROCEDURE 488
CREATE SEQUENCE 512
CREATE STOGROUP 515
CREATE SYNONYM 516
CREATE TABLE 517
CREATE TABLESPACE 534
CREATE TRIGGER 544
CREATE UNIQUE CONSTRAINT 473
CREATE VIEW 547
DROP ALIAS 549
DROP CHECK 549
DROP DATABASE 550
DROP FOREIGN KEY 551
DROP INDEX 552
DROP SEQUENCE 553
DROP STOGROUP 553
DROP SYNONYM 554
DROP TABLE 554
DROP TABLESPACE 555
DROP TRIGGER 555
DROP UNIQUE CONSTRAINT 550
DROP VIEW 556
elements 377
variables 377
CDLCHANGERULES ALUIN keyword 77, 98
CDLDSN installation option 140
CDLPS installation option 140
CDLSS installation option 140
CDLU installation option 140
CDNAME attribute
  modifying for a check constraint in a 
  REPLICATE WORKID statement 576
modifying for a column in a REPLICATE 
  WORKID statement 576, 577
modifying for a database in a REPLICATE 
  WORKID statement 577
modifying for a parm column in a REPLICATE 
  WORKID statement 579, 580
modifying for a relation in a REPLICATE 
  WORKID statement 581
modifying for a relation key in a REPLICATE 
  WORKID statement 580, 581
modifying for a routine in a REPLICATE 
  WORKID statement 581–585
modifying for a storage group in a REPLICATE 
  WORKID statement 585
modifying for a synonym in a REPLICATE 
  WORKID statement 586
modifying for a table in a REPLICATE 
  WORKID statement 586, 587
modifying for a table space in a REPLICATE 
  WORKID statement 588, 589
modifying for a table space partition in a 
  REPLICATE WORKID statement 589, 
  590
modifying for a view in a REPLICATE 
  WORKID statement 590, 591
modifying for an alias in a REPLICATE 
  WORKID statement 575
modifying for an index in a REPLICATE 
  WORKID statement 578, 579
modifying for an index key column in a 
  REPLICATE WORKID statement 579
modifying for an index partition in a 
  REPLICATE WORKID statement 589, 
  590

CDOWNER attribute
modifying for a check constraint in a 
  REPLICATE WORKID statement 576
modifying for a column in a REPLICATE 
  WORKID statement 576, 577
modifying for a database in a REPLICATE 
  WORKID statement 577
modifying for a parm column in a REPLICATE 
  WORKID statement 579, 580
modifying for a relation in a REPLICATE 
  WORKID statement 581
modifying for a relation key in a REPLICATE 
  WORKID statement 580, 581
modifying for a routine in a REPLICATE 
  WORKID statement 581–585
modifying for a storage group in a REPLICATE 
  WORKID statement 585
modifying for a synonym in a REPLICATE 
  WORKID statement 586
modifying for a table in a REPLICATE 
  WORKID statement 586, 587
modifying for a table space in a REPLICATE 
  WORKID statement 588, 589
modifying for a table space partition in a 
  REPLICATE WORKID statement 589, 
  590
modifying for a view in a REPLICATE 
  WORKID statement 590, 591
modifying for an alias in a REPLICATE 
  WORKID statement 575
modifying for an index in a REPLICATE 
  WORKID statement 578, 579
modifying for an index key in a REPLICATE 
  WORKID statement 579
modifying for an index partition in a 
  REPLICATE WORKID statement 589, 
  590

CFK (create foreign key) action code 32
CFK (create foreign key) command 27
CGT (create global temporary table) action code 32
CGT (create global temporary table) command 27
CH (changed object list) command 27
Change Definition Language. See CDL change flag keywords 45
CHANGE MANAGER
  installation option descriptions 159
  installation options, list of 159
  change rules 45
  CHANGE TABLE option, IMPORT command 302
  change-type change rule 45
  CHANGELVL attribute
modifying for a check constraint in a
  \texttt{REPLICATE WORKID} statement 576
modifying for a column in a \texttt{REPLICATE WORKID} statement 576, 577
modifying for a database in a \texttt{REPLICATE WORKID} statement 577
modifying for a parm column in a \texttt{REPLICATE WORKID} statement 579, 580
modifying for a relation in a \texttt{REPLICATE WORKID} statement 581
modifying for a relation key in a \texttt{REPLICATE WORKID} statement 580, 581
modifying for a routine in a \texttt{REPLICATE WORKID} statement 581–585
modifying for a storage group in a \texttt{REPLICATE WORKID} statement 585
modifying for a synonym in a \texttt{REPLICATE WORKID} statement 586
modifying for a table in a \texttt{REPLICATE WORKID} statement 586
modifying for a table space in a \texttt{REPLICATE WORKID} statement 586, 587
modifying for a table space partition in a \texttt{REPLICATE WORKID} statement 588, 589
modifying for a view in a \texttt{REPLICATE WORKID} statement 590
modifying for an alias in a \texttt{REPLICATE WORKID} statement 575
modifying for an index in a \texttt{REPLICATE WORKID} statement 578, 579
modifying for an index key in a \texttt{REPLICATE WORKID} statement 579
modifying for an index partition in a \texttt{REPLICATE WORKID} statement 589, 590

\textsc{changerulesin2 \texttt{aluin} keyword} 77

changes to the product 14

\textit{check attribute}

\begin{itemize}
  \item change rules for DB2 objects 45
  \item for SET and WHERE clauses in a view 616
  \item modifying for a view in a \texttt{REPLICATE WORKID} statement 590, 591
\end{itemize}

check constraint

\begin{itemize}
  \item DML keyword 559
  \item check constraints
\end{itemize}

attributes for SET and WHERE clauses 605

modifying attributes in a \texttt{REPLICATE WORKID} statement 576
performable actions 557
profile change rules 45

\textsc{check data utility, -\texttt{chek} worklist command} 305

\textsc{check lob utility, -\texttt{chek} worklist command} 305

\textsc{check plus utility, -\texttt{bmck} worklist command} 280

\textsc{checkcondition attribute} 605
checkConstraintName variable 377

\textsc{checkdopt \texttt{ajxin} keyword} 65

\textsc{checkdopt \texttt{pof} keyword} 206

\textsc{checkname attribute} 605

\textsc{checkopt \texttt{aixin} keyword} 57

\textsc{checks attribute} 613, 614

\textsc{checks dml keyword} 559

\textsc{checkman_load \texttt{pof} keyword} 206

\textsc{csize4k \texttt{aluin} keyword} 78

\textsc{cix (create index) action code} 32
\textsc{cix (create index) command} 27

\textsc{ck (check constraint) action code} 32

\textsc{ck object type} 19

\textsc{ckexceptions \texttt{aluin} keyword} 78, 142

\textsc{ckname attribute} 576

\textsc{cksegsize parameter, \texttt{override \texttt{aluin} keyword}} 112

\textsc{cleanup_rc \texttt{pof} keyword} 206

\textsc{clist macro} 332

\textsc{clname attribute}

\begin{itemize}
  \item modifying for a column in a \texttt{reuplicate workid} statement 576, 577
  \item modifying for an index key in a \texttt{reuplicate workid} statement 579
\end{itemize}

\textsc{clonedata installation option} 78, 100, 140, 166

\textsc{clonedata \texttt{aluin} keyword} 78, 140

\textsc{close attribute} 45, 50

\textsc{closerule attribute} 607, 608, 614, 615

\textsc{clust attribute} 578, 579

\textsc{clustering attribute} 607, 608

\textsc{clustertype attribute} 613, 614

\textsc{cm (comment) action code} 32

\textsc{cm (comment) command} 27

\textsc{cm/pilot keywords}
CM/PILOT 352
CMPIN1T 353
CMPIN2T 353
DEBUG 355
NOREPLACE 355
REPLACE 355
REUSEWORKID 352
SSID 355
TRIAL 352, 355

CM/PILOT worklist commands
-ANLY (Analysis) 352
-BJCL (Batch Execution JCL Generation) 353
-COMP (Compare) 353
-DML (DML section) 354
-IMP (Import) 354
-OPTS (Options) 355
-REPL (REPL section) 356
-TASK (task ID) 357

CM/PILOT, omitting objects with Unicode attributes 592

CMP installation option 140, 166
CMPDIAG installation option 140
CMPILOT ALUIN keyword 78
CMPILOT CM/PILOT keyword 352
CMPIN1 ALUIN keyword 78
CMPIN1T CM/PILOT keyword 353
CMPIN2 ALUIN keyword 78
CMPIN2T CM/PILOT keyword 353
CMPLIMIT1 installation option 140, 166
CMPLIMIT2 installation option 140, 166
CMPPROFILE ALUIN keyword 77
CMPSCWBF installation option 140, 166
CMPSCWBS installation option 167
CMPSCWBSinstallation option 140
CMPSPTXT installation option 130, 141, 167
CMPTYPE1 ALUIN keyword 79
CMPTYPE2 ALUIN keyword 79
CNTL_DATACLASS POF keyword 206
CNTL_EXPDT POF keyword 206
CNTL_MGMTCLASS 206
CNTL_PREFIX POF keyword 206
CNTL_PRIQTY POF keyword 207
CNTL_RETPD POF keyword 207
CNTL_SECQTY POF keyword 207
CNTL_STORCLASS POF keyword 207
CNTL_UNIT POF keyword 207
CNTLMOUT_DSN POF keyword 207
CNTLMSCH_DSN POF keyword 207
CO (column) action code 32
CO (column) command 27

COLCARD attribute 605, 606
COLCARDF attribute 605, 606
COLLECTION attribute 50
COLLID attribute
- for SET and WHERE clauses in a routine 610–612
- modifying for a routine in a REPLICATE WORKID statement 581–585

COL attribute 576, 577
COLNAME attribute
- change rules for application objects 50
- change rules for DB2 objects 45
- for SET and WHERE clauses in a foreign key 607
- for SET and WHERE clauses in an index key 608
- modifying for a relation key in a REPLICATE WORKID statement 580, 581

COLNO attribute
- for SET and WHERE clauses in a column 605, 606
- for SET and WHERE clauses in a field 606, 607
- for SET and WHERE clauses in a foreign key 607
- modifying for a column in a REPLICATE WORKID statement 576, 577
- modifying for an index key in a REPLICATE WORKID statement 579

COLSEQ attribute
- for SET and WHERE clauses in a foreign key 607
- for SET and WHERE clauses in an index key 608
- modifying for a relation key in a REPLICATE WORKID statement 580, 581

COLSNTXT attribute 590, 591
COLTYPE attribute
- for SET and WHERE clauses in a column 605, 606
- for SET and WHERE clauses in a field 606, 607
- for SET and WHERE clauses in a sequence 612
- modifying for a column in a REPLICATE WORKID statement 576, 577

columnName variable 377
columnNameList variable 377
columns
- attributes for SET and WHERE clauses 605, 606
- deleting 561
- DML keyword 559
- modifying attributes in a REPLICATE WORKID statement 576, 577
- performable actions 557
COLUMNS DML keyword 559
Command line commands, list of 26
commands
  -BMCU (BMCU Execute a BMC Utility) 357
  -DBUG (Debug) 357
  -SQLM (SQL Statement) 361
comment lines 261
COMMIT ALUIN keyword 79
COMMIT_ON_RETURN attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE
  WORKID statement 581–585
Compare CM/PILOT worklist command 353
COMPRESS attribute
  change rule for DB2 objects 45
  change rules for application objects 50
  for SET and WHERE clauses in a table space
  partition 615
  for SET and WHERE clauses in an index 607, 608
  modifying for a table space partition in a
  REPLICATE WORKID statement 589, 590
  modifying for an index in a REPLICATE
  WORKID statement 578, 579
CONCUR_ACC_RES attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE
  WORKID statement 581–585
constant variable 377
constraintName variable 377
constraintText variable 377
control card, NULLTYPE T1 NULLCHAR ? 275
conventions, documentation 12
converting tables 597
converting to table-controlled partitioning 597
COPY attribute 607, 608
COPY utility 307
COPY01 AJXIN keyword 65, 141
COPY02 AJXIN keyword 65, 141
COPYDD01 installation option 65, 167
COPYDD02 installation option 65, 167
COPYDDN ALUIN keyword 79
COPYPARTS parameter 315
CPLDIAG installation option 141
CPLWDSN installation option 141
CPYEXP_DATACLASS POF keyword 208
CPYEXP_EXPDT POF keyword 208
CPYEXP_MGMTCLASS POF keyword 208
CPYEXP_PREFIX POF keyword 208
CPYEXP_INDEX POF keyword 208
CPYEXP_STORCLASS POF keyword 208
CPYEXP_SUPPRESS_SUFF POF keyword 208
CPYEXP_UNIT POF keyword 208
CR (create) command 27
CR AL (create alias) command 26
CR CK (create check constraint) command 27
CR DB (create database) command 27
CR FK (create foreign key) command 27
CR GT (create global temporary table) command 27
CR IX (create index) command 27
CR SY (create synonym) command 27
CR symbolic variable 146
CR TB (create table) command 27
CR TR (create trigger) command 28
CR TS (create table space) command 28
CR UC (create unique constraint) command 28
CR VW (create view) command 28
CRAUTH attribute
  DELETE SYNTABLE statement 563
  DELETE WORKID statement 565
CRDATE attribute
  DELETE SYNTABLE statement 563
  DELETE WORKID statement 565
CREATE ALIAS statement 470
CREATE AUXILIARY TABLE statement 471
CREATE CHECK statement 472
CREATE DATABASE statement 474
CREATE FOREIGN KEY statement 477
change rules for application objects 50
change rules for DB2 objects 45
for SET and WHERE clauses in a check
constraint 605
for SET and WHERE clauses in a database 606
for SET and WHERE clauses in a foreign key
607
for SET and WHERE clauses in a relation 610
for SET and WHERE clauses in a storage group
613
for SET and WHERE clauses in a synonym 613
for SET and WHERE clauses in a table 613, 614
for SET and WHERE clauses in a table space
614, 615
for SET and WHERE clauses in a view 616
for SET and WHERE clauses in an alias 604, 605
for SET and WHERE clauses in an index 607,
608
CRTIME attribute
DELETE SYNTABLE statement 563
DELETE WORKID statement 565
CSG (create storage group) command 27
CSP (create stored procedure) command 27
CSQ (create sequence) command 27
CSY (create synonym) action code 32
CSY (create synonym) command 27
CT (constraint text) action code 32
CTB (create table) action code 32
CTB (create table) command 27
CTC (create clone table) action code 32
CTC (create clone table) command 28
CTR (create trigger) action code 32
CTR (create trigger) command 28
CTS (create table space) action code 32
CTS (create table space) command 28
CUC (create unique constraint) action code 32
CUC (create unique constraint) command 28
CVALOFF ALUIN keyword 80
CVW (create view) action code 32
CVW (create view) command 28
CYCLE attribute 612

D
D (delete) action code 32
D (drop) action code 32
DA symbolic variable 146
DA TEMPLATE descriptor variable 146
DASD_LOAD POF keyword 209
DASDDOPT AEXIN keyword 57
DASDDOPT AJXIN keyword 66
DASDDOPT POF keyword 209
DASDMAN installation option 141, 167
DASDTRIG AEXIN keyword 57
data sets
generation data group 308
SYSMAP 326
WRK100n 293
DATA_PACKER_LOAD POF keyword 209
data-only migration 321, 340
DATABASE attribute 45, 50
databaseName variable 377
databases
attributes for SET and WHERE clauses 606
deleting 561
DML keyword 559
modifying attributes in a REPLICATE WORKID statement 577
performable actions 557
profile change rules 45
DATABASES DML keyword 559
DATACAPTURE attribute
change rules for application objects 50
change rules for DB2 objects 45
for SET and WHERE clauses in a table 613, 614
modifying for a table in a REPLICATE WORKID statement 586, 587
DATACLAS attribute 613
DATACLAS installation option 141, 167
DATACLASS attribute 589, 590
DATASETSIZING AJXIN keyword 66
DATASETSIZING POF keyword 209
dataType Name variable 377
dataTypeSchema variable 377
DATAWK_NBR POF keyword 209
DATAWK_UNIT POF keyword 210
DATE installation option 141, 168
DATE OUTPUT descriptor variable
DATE symbolic variable 146
DT symbolic variable 147
JYMD symbolic variable 150
YMD symbolic variable 156
DATE symbolic variable 146
DATE TEMPLATE descriptor variable
DATE symbolic variable 146
JYMD symbolic variable 150
YMD symbolic variable 156
DATE_FORMAT attribute
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE WORKID statement 581–585
DATEJ symbolic variable 146
DAY OUTPUT descriptor variable 146
DAY symbolic variable 146
DAY TEMPLATE descriptor variable 146
DB (database) action code 32
DB object type 19
DB OUTPUT descriptor variable
CR symbolic variable 146
DB symbolic variable 146
DBNAME symbolic variable 146
IXCR symbolic variable 148
TBCR symbolic variable 153
TBCRE symbolic variable 153
VCAT symbolic variable 155
DB symbolic variable 146
DB TEMPLATE descriptor variable
CR symbolic variable 146
DB symbolic variable 146
DBNAME symbolic variable 146
IXCR symbolic variable 148
TBCR symbolic variable 153
TBCRE symbolic variable 153
VCAT symbolic variable 155
DB_Generated_DocID_for_xml column 322, 346
DB..IS TEMPLATE descriptor variable 148
DB..SN TEMPLATE descriptor variable 151
DB..TS TEMPLATE descriptor variable 153
DB..TS OUTPUT descriptor variable
IXNODE symbolic variable 148
OBNAM symbolic variable 151
OBNOD symbolic variable 151
TBNODE symbolic variable 153
DB2 command, -CMD worklist command 307
DB2 object types, abbreviations 19
DB2 utilities
CHECK DATA 305
CHECK LOB 305
COPY 307
LOAD 321
REBUILD 329
REORG 330
RUNSTATS 333
DB2 Utilities
DSN1COPY 358
DB2EXIT POF keyword 210
DB2LOAD POF keyword 210
DB2STATSUPD ALUIN keyword 80, 145
DB2STMSGS AEXIN keyword 58
DB2V2 symbolic variable 146
DB2V3 symbolic variable 147
DBID attribute 615, 616
DBINFO attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
DBNAME attribute
  for SET and WHERE clauses in a table 613, 614
  for SET and WHERE clauses in a table space 614, 615
  for SET and WHERE clauses in a table space partition 615
  modifying for a database in a REPLICATE WORKID statement 577
  modifying for a table in a REPLICATE WORKID statement 586, 587
  modifying for a table space in a REPLICATE WORKID statement 588, 589
  modifying for a table space partition in a REPLICATE WORKID statement 589, 590
  modifying for an index partition in a REPLICATE WORKID statement 589, 590
DBNAME symbolic variable 146
DBOWNER attribute 577
DBRM1 AJXIN keyword 66, 141
DBRM1 installation option 66, 141
DBRM2 AJXIN keyword 66, 141
DBRM2 installation option 66, 141
DBRM3 AJXIN keyword 66, 141
DBRM3 installation option 66, 141
DBRMLIB ALUIN keyword 80, 141
DBRMLIB attribute 50
DBRMLIB installation option 81, 141, 168
DD (block delete) action code 32
DD (delete) action code 32
DDD symbolic variable 147
DDNAME parameter
  used in -UNRC worklist command 348
DDNAME symbolic variable 147
DDOPT symbolic variable 147
DDSEQ symbolic variable 147
DEBUG ALUIN keyword 81
DEBUG CM/PILOT keyword 355
d debug worklist command 357
DEBUG_MODE attribute 50
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
DEBUG(INIT) ALUIN keyword 81
DEBUGUNLD AEXIN keyword 58
DEC31 attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
DEC31_SCALE attribute
  modifying for a routine in a REPLICATE WORKID statement 581–585
DEF_GDG_BASE POF keyword 210
DEF_GDG_LIMIT POF keyword 210
DEF_GDG_NOSCR POF keyword 210
DEF_GDG2_LIMIT POF keyword 210
default options. See installation option
DEFAULTOFF ALUIN keyword 82
DEFAULTVALUES attribute 605, 606
DEFERPREP attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
DEFERPREPARE attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
DEFERUIX installation option 82, 141, 168
DEFERUNIQUEIX ALUIN keyword 82, 141
DEFGDGBASE AJXIN keyword 66
DEFGDGLIMIT AJXIN keyword 66
DEFINE attribute
  change rules for application objects 50
  change rules for DB2 objects 45
  for SET and WHERE clauses in a table space 614, 615
  for SET and WHERE clauses in an index 607, 608
DEGREE attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
DELETE statement 561
DELETE SYNC TABLE statement 563
DELETE SYNC TABLE statement attributes 563
DELETE WORKID statement 565
DELETE WORKID statement attributes 565
DELETEAGE AJXIN keyword 66
DELETEAGE ALUIN keyword 82
DELETEFILES NO parameter 288
DELETERULE attribute
for SET and WHERE clauses in a foreign key 607
for SET and WHERE clauses in a relation 610
modifying for a relation in a REPLICATE WORKID statement 581
deleting
columns 561
databases 561
objects 557
sync table entries 563
work IDs 565
DETERMINISTIC attribute
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE WORKID statement 581–585
DEVTYPE attribute
change rules for application objects 50
change rules for DB2 objects 45
modifying for a table space partition in a
REPLICATE WORKID statement 589, 590
modifying for an index partition in a
REPLICATE WORKID statement 589, 590
DIAG_MSGCLASS POF keyword 211
DISC_DATACLASS POF keyword 211
DISC_DATACLASS_ALT POF keyword 211
DISC_MGMTCLASS POF keyword 211
DISC_MGMTCLASS_ALT POF keyword 211
DISC_PREFIX POF keyword 211
DISC_PRIQTY POF keyword 211
DISC_RETLPD POF keyword 211
DISC_SECQTY POF keyword 212
DISC_STORCLASS POF keyword 212
DISC_STORCLASS_ALT POF keyword 212
DISC_THRESH POF keyword 212
DISC_UNIT POF keyword 212
DISC_UNIT_ALT POF keyword 212
DISCARDS ALUIN keyword
description 83
DISCARDS installation option 141
dynamically allocated discard data sets 288
DISCARDS installation option 83, 141, 168
DISP_ALLOW_POPUP POF keyword 212
DISP_AUTO_TAB POF keyword 213
DISP_LOCATION POF keyword 213
DISP_OMIT_CHAR POF keyword 213
DISP_STATS POF keyword 213
DISP_VAR_DBUG POF keyword 213
DML attributes, specifying to change in a LIKE statement 560
attributes, specifying to change in a
REPLICATE WORKID statement 573
catalog indirection 557
catalog tables, searching 559
catalog tables, updating 559
changing objects 557
deleting objects 557
dependent objects, including in migrate
execution 559
importing 558
INCLUDE clause of MIGRATE statement 568
keyword to catalog table mapping 559
keywords, object-type 559
liking objects 557
migrating objects 557
object types 559
overview 557
performable actions 557
processing 559
statements 558
syntax diagrams 558
updating objects 557
DML keywords, list of 559
DML section, -DML CM/PILOT worklist
command 354
DML SET clause
catalog table columns, using 604
described for LIKE statement 560
LIKE statement, use in 560
syntax for LIKE statement 560
values, specifying for LIKE statement 560
DML statements
DELETE 561
DELETE SYNC_TABLE 563
DELETE WORKID 565
JOIN keyword 602
LIKE 566
MIGRATE 568
REPLICATE WORKID 573
SET OMITUNICODE 592
SET PRESERVELIMITKEY 593
SET SPACE ESTIMATION 594
UPDATE 597
DML WHERE clause
improving performance 602
JOIN keyword 602
documentation information
DOPTS command
DROP ALIAS statement
DROP CHECK statement
DROP DATABASE statement
DROP FOREIGN KEY statement
DROP INDEX statement
DROP SEQUENCE statement
DROP STOGROUP statement
DROP SYNONYM statement
DROP TABLE statement
DROP TABLESPACE statement
DROP TRIGGER statement
DROP UNIQUE CONSTRAINT statement
DROP VIEW statement
DROPALL (drop all) command
DROPRESTRICT attribute
change rules for DB2 objects
modifying for a table in a REPLICATE WORKID statement
DSN parameter, used in -COPY command
DSN1COPY utility
DSNAME parameter
used in -BMCC worklist command
used in -BMCD worklist command
used in -BMCV worklist command
used in -UNRC worklist command
DSNCHECK44 POF keyword
DSNHLQ AJXIN keyword
DSNTIAD_PLAN POF keyword
DSNUM OUTPUT descriptor variable
DSNUM symbolic variable
DSSIZE attribute
change rules for DB2 objects
for SET and WHERE clauses in a table space
DSCAT attribute
DT symbolic variable
DT TEMPLATE descriptor variable
DUAL installation option
See also STOPICP
DUAL installation option
DV (default value) action code
DV (default value) command
DYNCOPY ALUIN keyword
description
DYNCOPY installation option
UTILCOPY ALUIN keyword
DYNCOPY installation option
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY installation option
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY installation option
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY installation option
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNWORKUNIT AEXIN keyword
E
E (edit comment) action code
E (edit) action code
EAP installation option
EDITPROC attribute
EDPROC attribute
EIP installation option
EMPTYCARD ALUIN keyword
ENABLE attribute
ENCODING_CCSID attribute
ENCODING_SCHEME attribute
for SET and WHERE clauses in a routine
for SET and WHERE clauses in a database
for SET and WHERE clauses in a parm
for SET and WHERE clauses in a table
for SET and WHERE clauses in a table space
END command
using the -UNRC worklist command
using the IBM COPY utility
using the IBM UNLOAD utility
using the NGT Copy utility
using the NGT Recover utility
using the UNLOAD PLUS utility
DYNAMICRULES attribute
for SET and WHERE clauses in a routine
modifying for a routine in a REPLICATE WORKID statement
DYNCOPY ALUIN keyword
description
DYNCOPY installation option
UTILCOPY ALUIN keyword
DYNCOPY installation option
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY installation option
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY installation option
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY installation option
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNCOPY ALUIN keyword
NODYNCOPY ALUIN keyword
overriding keywords
DYNWORKUNIT AEXIN keyword
E
E (edit comment) action code
E (edit) action code
EAP installation option
EDITPROC attribute
EDPROC attribute
EIP installation option
EMPTYCARD ALUIN keyword
ENABLE attribute
ENCODING_CCSID attribute
ENCODING_SCHEME attribute
for SET and WHERE clauses in a routine
for SET and WHERE clauses in a database
for SET and WHERE clauses in a parm
for SET and WHERE clauses in a table
for SET and WHERE clauses in a table space
END command
end global commands worklist command 265, 309
end parallel processing worklist command 265, 310
end unit of work worklist command 267, 310
ENFORCE attribute 581
ENFORCED attribute 610
ENV AEXIN keyword 58
ENV ALUIN keyword 85
ENVI command 28
ENVIRONMENT command 28
ENVP installation option 141, 169
EPOCH attribute 615
EPP installation option 141, 169
EPROC attribute 586, 587
ERASE attribute 45, 50
ERASERULE attribute 607, 608, 614, 615
ERR_DATACLASS POF keyword 214
ERR_DATACLASS_ALT POF keyword 214
ERR_EXPDT POF keyword 214
ERR_MGMTCLASS POF keyword 214
ERR_MGMTCLASS_ALT POF keyword 214
ERRPREFIX POF keyword 214
ERR_PRIQTY POF keyword 215
ERR_RETPD POF keyword 215
ERR_SECQTY POF keyword 215
ERR_STORCLASS POF keyword 215
ERR_STORCLASS_ALT POF keyword 215
ERR_THRESH POF keyword 215
ERR_UNIT POF keyword 215
ERR_UNIT_ALT POF keyword 215
error worklist command 310
establishing a baseline, -BASE CM/PILOT worklist
command 263
estimating space
explicitly 594
implicitly 594
overriding 594
using the SET SPACE ESTIMATION statement 594
EURO AJXIN keyword 67, 142
EURO ALUIN keyword 86, 142
EURO installation option 67, 169
EURO ALUIN keyword 86
overriding keyword 142
EVENTS AEXIN keyword 58
EXACT parameter, SPTEXT ALUIN keyword 130
examples
BMCU command 357
DSN1 command 358
SQLM command 361
EXCEPT installation option 78, 142, 170
EXEC_LOAD POF keyword 216
EXPLAIN attribute
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE WORKID statement 581–585
EXPMAXTS ALUIN keyword 87
EXPORT option, EXPORT command 271
external SQL stored procedures worklist command 338
EXTERNAL_NAME attribute
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE WORKID statement 581–585
EXTERNAL_SECURITY attribute
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE WORKID statement 581–585

F

FAULT attribute 605, 606
FCMD symbolic variable 147
FCPY_DATACLASS POF keyword 216
FCPY_EXPDT POF keyword 216
FCPY_MGMTCLASS POF keyword 216
FCPY_PREFIX POF keyword 216
FCPY_PRIQTY POF keyword 216
FCPY_RETPD POF keyword 216
FCPY_SECQTY POF keyword 217
FCPY_STORCLASS POF keyword 217
FCPY_SUPPRESS_SUFF POF keyword 217
FCPY_UNIT POF keyword 217
FENCED attribute
for SET and WHERE clauses in a routine 610–612
FEP installation option 142, 170
FI (find) command 29
FIELDPROC attribute 45, 50
fields
attributes for SET and WHERE clauses 606, 607
DML keyword 559
performable actions 557
FIELDS DML keyword 559
FILEREF parameter 348
FILT_DATACLASS POF keyword 217
FILT_EXPDT POF keyword 217
FILT_MGMTCLASS POF keyword 217
FILT_PREFIX POF keyword 217
FILT_PRIQTY POF keyword 218
FILT_RETPD POF keyword 218
FILT_SECQTY POF keyword 218
FILT_STORCLASS POF keyword 218
FILT_UNIT POF keyword 218
FIND command 29
FK (foreign keys) action code 32
FK object type 19
FKKEYCARD attribute 578, 579
FKNAME attribute 580, 581
FKNN parameter, OVERRIDE ALUIN keyword 112
FKNO attribute 580, 581
FLDPROC attribute 605–607
FLOW AEXIN keyword 58
FOR_UPDATE_CLAUSE attribute
  modifying for a routine in a REPLICATE WORKID statement 581–585
force-type change rule 45
FORCECHECK ALUIN keyword 87
FORCERESMAX ALUIN keyword 88
foreign keys
  attributes for SET and WHERE clauses 607
  DML keyword 559
  performable actions 557
  profile change rules 45
FOREIGN KEYS DML keyword 559
FOREIGNKEY attribute 605, 606
FORMAT BMCLOAD 275
FORMAT BMCUNLOAD 281
FP (field proc parameters) action code 32
FP (field proc) command 29
FPROC attribute 576, 577
FREEPAGE attribute
  change rules for application objects 50
  change rules for DB2 objects 45
  for SET and WHERE clauses in a table space partition 615
  for SET and WHERE clauses in an index partition 608, 609
  modifying for a table space partition in a
    REPLICATE WORKID statement 589, 590
  modifying for an index partition in a
    REPLICATE WORKID statement 589, 590
G
"G" (generate CDL) action code 32
GBPCACHE attribute
  change rules for DB2 objects 45
  for SET and WHERE clauses in a table space partition 615
  for SET and WHERE clauses in an index partition 608, 609
  modifying for a table space in a REPLICATE WORKID statement 589, 590
  modifying for an index partition in a
    REPLICATE WORKID statement 589, 590
GDG symbolic variable 147
GDG_MODEL POF keyword 218
GDG(generation data group) 308
GENERATED ALWAYS, defined for identity column 292
GENERATED ALWAYS, ROWID column
  -BMCB worklist command 269
  -BMCD worklist command 278
  -BMCL worklist command 287
  -LOAD worklist command 323
  -UNLI worklist command 348
GENERATED BY DEFAULT, ROWID column
  -BMCB worklist command 269
  -BMCD worklist command 278
  -BMCL worklist command 287
  -LOAD worklist command 323
  -UNLI worklist command 348
generation data group(GDG) 308
GENOBID ALUIN keyword 91, 102, 142
GENOBID installation option 91, 102, 142, 170
GENOBID parameter, OVERRIDE ALUIN keyword 112, 142
GLAUTHID ALUIN keyword 91, 142
GLID AJXIN keyword 67
GLID installation option 91, 142
Global Authorization ID worklist command 311
GRANTEE attribute 45, 50
GRANTOR attribute 45, 50
GRANULARITY attribute 615, 616
GROUP_MEMBER attribute 606
GRPNM symbolic variable 147
GT (global temporary table) action code 32
H

HASHFAIL AEXIN keyword 58
HASHFAIL POF keyword 218
HASHWARNRC AEXIN keyword 58
HASHWARNRC POF keyword 218
HELP command 29
HI command 29
HIDDEN attribute 605, 606
HISTORY ALL parameter 333
HISTORY NONE parameter 333
HISTORY ALL ALUIN keyword 92, 144
HLQ.BMCCNTL data set, ACMDMLF1 592
HLQ.UBMCCNTL members
  product options file 181
HM symbolic variable 147
HMS symbolic variable 147
HO symbolic variable 147
HO TEMPLATE descriptor variable 147
HO.MI TEMPLATE descriptor variable
  HM symbolic variable 147
  TIME4 symbolic variable 153
HOUR OUTPUT descriptor variable 147
HOUR symbolic variable 147
HOUR.MINUTE OUTPUT descriptor variable
  HM symbolic variable 147
  TIME4 symbolic variable 153
HSMVOL ALUIN keyword 92, 142
HSMVOL installation option 170
  HSMVOL ALUIN keyword 92
  NOHSMVOL ALUIN keyword 102
  overriding keywords 142
IBMREORG ALUIN keyword 92, 143
IBMUNLOAD ALUIN keyword 92
IC symbolic variable 148
IC TEMPLATE descriptor variable
  IC symbolic variable 148
  ICTYPE symbolic variable 148
  JOBTYP symbolic variable 149
  OBJT symbolic variable 151
  OBJTYP symbolic variable 151
  RTYPE symbolic variable 152
  RUNTYP symbolic variable 152
  TYPE symbolic variable 154
ICP (index-controlled partitioning) command 29
ICTYPE OUTPUT descriptor variable 148
ICTYPE symbolic variable 148
ICTYPE TEMPLATE descriptor variable 148
ID (identity column) action code 32
ID command 29
IDAAIGNORE ALUIN keyword 93
IDCAMS execution worklist command 261
IDCAMS utility 261
identifier variable 377
IDENTITY attribute 45
identity columns 292
IDENTITYOVERRIDE YES parameter, BMC
  LOADPLUS 292
IMP installation option 142, 170
IMPDIAG AJXIN keyword 67
IMPDIAG installation option 142
IMPLICIT attribute 588, 589
IMPLICIT parameter 594
IMPMIGPROFILE AJXIN keyword 67
IMPORT ALUIN keyword 94
importing DML 558
IMPORTAUTH attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
IMPORTDATE attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
importing DML 558
IMPORTTIME attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
INCLUDE ALUIN keyword 94
INCLUDE_SYSPRIN2 POF keyword 219
include-type change rule 45
INCLUDEDPARENTS parameter, OVERRIDE ALUIN keyword 112
INCREMENT attribute 612
index keys
attributes for SET and WHERE clauses 608
DML keyword 559
modifying attributes in a REPLICATE WORKID statement 579
performable actions 557
index partitions
attributes for SET and WHERE clauses 608, 609
DML keyword 559
modifying attributes in a REPLICATE WORKID statement 589, 590
performable actions 557
index-controlled partitioning, converting to table-controlled partitioning 597
INDEXBP attribute 606
indexes
attributes for SET and WHERE clauses 607, 608
DML keyword 559
modifying attributes in a REPLICATE WORKID statement 578, 579
performable actions 557
profile change rules 45
INDEXES DML keyword 559
INDEXES option, IMPORT command 302
indexName variable 377
indexOwner variable 377
INDEXPARTS DML keyword 559
INDEXTYPE attribute 578, 579
INLINE attribute 45
installation options 53, 139
IOALOAD1 POF keyword 219
IOALOAD2 POF keyword 219
IPPARTS parameter 317
is LOB column empty worklist command 313
IS symbolic variable 148
is table empty worklist command 312
IS TEMPLATE descriptor variable 148
ISOLATION attribute
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE WORKID statement 581–585
ITERATIONMODE AEXIN keyword 58
IX (indexes) action code 32
IX object type 19
IX symbolic variable 148
IXA object type 19
IXC object type 19
IXCRA symboic variable 148
IXCT object type 19
IXG object type 19
IXGC object type 19
IXGT object type 19
IXN object type 19
IXNAME attribute
for SET and WHERE clauses in a relation 610
for SET and WHERE clauses in an index key 608
for SET and WHERE clauses in an index partition 608, 609
modifying for a relation in a REPLICATE WORKID statement 581
modifying for a table space in a REPLICATE WORKID statement 589, 590
modifying for an index in a REPLICATE WORKID statement 578, 579
modifying for an index key in a REPLICATE WORKID statement 579
modifying for an index partition in a REPLICATE WORKID statement 589, 590
IXNAME symbolic variable 148
IXNEW attribute 579, 589, 590
IXNODE symbolic variable 148
IXOWNER attribute
for SET and WHERE clauses in a relation 610
modifying for a relation in a REPLICATE WORKID statement 581
modifying for a table space partition in a REPLICATE WORKID statement 589, 590
modifying for an index in a REPLICATE WORKID statement 578, 579
modifying for an index key in a REPLICATE WORKID statement 579
modifying for an index partition in a REPLICATE WORKID statement 589, 590
IXPADDEDN parameter, OVERRIDE ALUIN keyword 112
IXPADDEICY parameter, OVERRIDE ALUIN keyword 112
IXPC object type 19
IXPCT object type 19
IXPT object type 19
IXR object type 19
IXRC object type 19
IXRCT object type 19
IXRT object type 19
IXS object type 19
IXSPC symbolic variable 148
IXST object type 19
IXSU object type 19
IXT object type 19
IXU object type 19
IXUC object type 19
IXUCT object type 19
IXUT object type 19
IXV object type 19
IXVD object type 19
IXVN object type 19
IXVU object type 19
IXW object type 19
IXWC object type 19

J

JAR_ID attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
JARSCEMMA attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
JAVA_SIGNATURE attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
JCL generation, SYSMAP data set 326
JCLCLEANUP POF keyword 220
JCLDEBUG AJXIN keyword 67
JCLDEBUG command 29
JCLLIB POF keyword 220
JD symbolic variable 148
JDATE OUTPUT descriptor variable
DATEJ symbolic variable 146
JDATE symbolc variable 148
JU symbolic variable 149
JUL4Y symbolic variable 149
JULIAN symbolic variable 150
YYDDD symbolic variable 156
YYYYDDD symbolic variable 157
JDATE symbolic variable 148
JDATE TEMPLATE descriptor variable
  DATEJ symbolic variable 146
  JDATE symbolic variable 148
  JU symbolic variable 149
  JUL4Y symbolic variable 149
  JULIAN symbolic variable 150
  YYDDD symbolic variable 156
  YYYYDDD symbolic variable 157
JDAY OUTPUT descriptor variable
  DDD symbolic variable 147
  JDAY symbolic variable 148
  JJULD symbolic variable 148
  JDAY symbolic variable 148
  JDAY TEMPLATE descriptor variable
    DDD symbolic variable 147
    JDAY symbolic variable 148
    JJULD symbolic variable 148
    JDAY symbolic variable 148
    JDDN symbolic variable 148
    JDSN installation option 142
    JDSNB installation option 142
    JDSNBG installation option 142
    JDSNBR installation option 142
    JDSNC installation option 142
    JDSNCPL installation option 142
    JDSNE installation option 142
    JDSNI installation option 142
    JES3 POF keyword 220
    JHMS symbolic variable 148
    JJULD symbolic variable 148
    JOB_INCLUDE_MEMBER POF keyword 220
    JOBACPS AJXIN keyword 67
    JOBACSS AJXIN keyword 67
    JOBCARD1 POF keyword 220
    JOBCARD2 POF keyword 220
    JOBCARD3 POF keyword 220
    JOBCARD4 POF keyword 220
    JOBCARD5 POF keyword 220
    JOBCHAR symbolic variable 148
    JOBLOCK AJXIN keyword 142
    JOBNAME OUTPUT descriptor variable

Index 639
JOBCHEAR symbolic variable 148
JOBNAME symbolic variable 149
JPCOD symbolic variable 149
MEMBER symbolic variable 150
MEMBR symbolic variable 150
PGMR symbolic variable 152
WKOWN symbolic variable 156
WKOWNER symbolic variable 156
JOBNAME symbolic variable 149
JOBNAME TEMPLATE descriptor variable
ATTACH symbolic variable 146
DDOPT symbolic variable 147
JDDN symbolic variable 148
JOBCHAR symbolic variable 148
JOBNAME symbolic variable 149
JPCOD symbolic variable 149
MEMBER symbolic variable 150
MEMBR symbolic variable 150
PGMR symbolic variable 152
WKOWN symbolic variable 156
WKOWNER symbolic variable 156
ZACCTNUM symbolic variable 157
JOBSEQTYP AJXIN keyword 67
JOBTYP symbolic variable 149
JOIN DML keyword 602
JPCOD symbolic variable 149
JQID symbolic variable 149
JS1 symbolic variable 149
JS2 symbolic variable 149
JS4 symbolic variable 149
JSSID symbolic variable 149
JU symbolic variable 149
JUL4Y symbolic variable 149
JULIAN symbolic variable 150
JULIANDATE AJXIN keyword 67
JYMD symbolic variable 150

KEYCARD ALUIN keyword 94
NOKEYCARD ALUIN keyword 103
KEYCOLUMNS DML keyword 559
KEYLEN parameter 317
KEYSEQ attribute 605, 606
keywords
AEXIN 54
AEXPIN 54
AJXIN 54
AJXPOFIN 54
ALUIN 54
KY (index keys) action code 32
KY (index keys) command 30

L
L (like) action code 32
L (list) action code 32
LABEL attribute
for SET and WHERE clauses in a column 605, 606
for SET and WHERE clauses in a table 613, 614
for SET and WHERE clauses in a view 616
for SET and WHERE clauses in an alias 604, 605

LANGUAGE attribute
for SET and WHERE clauses in a routine 610–612
language elements, overview 362
Last parameter 29
LDSNUM symbolic variable 150
LENGTH attribute 605–607
for SET and WHERE clauses in a parm 609, 610
modifying for a parm column in a REPLICATE
WORKID statement 579, 580
LENGTH2 attribute 605, 606
LI symbolic variable 150
LI TEMPLATE descriptor variable 150
LIKE statement 566
LIKE TABLES statement, using TCPART 597
liking objects 557
LIMITKEY attribute 45
line commands, Specification 26
LINES AEXIN keyword 59
LIST symbolic variable 150
LIKE TABLES statement, using TCPART 597
LISTDEF symbolic variable 150
LISTDEF_DSN POF keyword 220
LIX (list indexes) action code 32
LIX (list indexes) command 30

K
KEEP-ALL-PRIMARY parameter, OVERRIDE
ALUIN keyword 113, 144
KEEPDYNAMIC attribute
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE
WORKID statement 581–585
KEYCARD ALUIN keyword 94, 103, 142
KEYCARD installation option 170
LK (limit key) action code 32
LK (limit key) command 30
LKVALID installation option 94, 103, 142, 171
LL_CLIB POF keyword 220
LL_CLIB2 POF keyword 220
LL_CLIB3 POF keyword 220
LL_CLIB4 POF keyword 220
LL_CLIB5 POF keyword 220
LL_LINK POF keyword 221
LL_LINK2 POF keyword 221
LL_LINK3 POF keyword 221
LL_LINK4 POF keyword 221
LL_LINK5 POF keyword 221
LL_MLIB POF keyword 221
LL_MLIB2 POF keyword 221
LL_MLIB3 POF keyword 221
LL_MLIB4 POF keyword 221
LL_MLIB5 POF keyword 221
LL_NAUTO POF keyword 221
LL_NAUTO2 POF keyword 221
LL_NAUTO4 POF keyword 221
LL_NAUTO5 POF keyword 221
LL_PLIB POF keyword 221
LL_PLIB2 POF keyword 221
LL_PLIB3 POF keyword 221
LL_PLIB4 POF keyword 221
LL_PLIB5 POF keyword 221
LL_SLIB POF keyword 221
LL_SLIB2 POF keyword 221
LL_SLIB3 POF keyword 221
LL_SLIB4 POF keyword 221
LL_SLIB5 POF keyword 221
LL_TLIB POF keyword 221
LL_TLIB2 POF keyword 221
LL_TLIB3 POF keyword 221
LL_TLIB4 POF keyword 221
LL_TLIB5 POF keyword 221
LL_XML POF keyword 222
LL_XML2 POF keyword 222
LL_XML3 POF keyword 222
LL_XML4 POF keyword 222
LL_XML5 POF keyword 222
LLQ POF keyword 222
LLQ symbolic variable 150
LOAD DATA REPLACE INTO TABLE syntax 88
LOAD DATA RESUME YES INTO TABLE
  REPLACE syntax 88
LOAD DATA RESUME YES INTO TABLE syntax 88
LOAD utility, -LOAD worklist command 321
LOADDOPT AJXIN keyword 67
LOADDOPT POF keyword 222
loading data 281
LOADDOPT AEXIN keyword 59
LOADDOPT AEXIN keyword 59
LOADDOPT POF keyword 222
LOADDOPT AJXIN keyword 67
LOADPLUS utility
  -BMCL worklist command 280
LOB data
  loading with BMC LOADPLUS 287
  loading with IBM LOAD 322
  unloading with BMC UNLOAD PLUS 279
  unloading with IBM UNLOAD 346
LOBCOL parameter
  -BMCC worklist command 272
  -COPY worklist command 309
  -LCMD worklist command 318
  -RNST worklist command 333
LOBGT32M ALUIN keyword 95
LOBTBN parameter
  -BMCC worklist command 272
  -COPY worklist command 309
  -LCMD worklist command 318
  -RNST worklist command 333
LOBTBO parameter
  -BMCC worklist command 272
  -COPY worklist command 309
  -LCMD worklist command 318
  -RNST worklist command 333
LOCATION attribute
  alias object 604, 605
  change rules for application objects 50
  change rules for DB2 objects 45
LOCATION installation option 142
LOCATION1 ALUIN keyword 95
LOCATION2 ALUIN keyword 95
locationID variable 377
LOCATOR attribute
  for SET and WHERE clauses in a parm 609, 610
  modifying for a parm column in a REPLICATE
    WORKID statement 579, 580
LOCK AEXIN keyword 142
LOCK installation option 142
LOCKMAX attribute
  change rules for application objects 50
  change rules for DB2 objects 45
  for SET and WHERE clauses in a table space 614, 615
  modifying for a table space in a REPLICATE
    WORKID statement 588, 589
LOCKPART attribute 614, 615
LOCKRULE attribute 614, 615
LOCKSIZE attribute 45, 50
LOCREM symbolic variable 150
LOCREM TEMPLATE descriptor variable 150
LOG ALUIN keyword 95, 142
LOG attribute
  change rules for DB2 objects 45
  for SET and WHERE clauses in a table space 614, 615
LOG installation option 95, 142, 171
LOGWK_NBR POF keyword 222
LOGWK_UNIT POF keyword 222
LONGVARCOLS parameter, OVERRIDE ALUIN keyword 113
L_PART OUTPUT descriptor variable 150
L_PART symbolic variable 150
L_R symbolic variable 150
L_R TEMPLATE descriptor variable 150
LTB (list tables) action code 32
LTB (list tables) command 30
LTS (list table spaces) action code 32
LTS (list table spaces) command 30

M
M (maintain change rules) action code 32
M (migrate) action code 32
MAINT command 30
MAINTENANCE attribute 613, 614
MAP_DATACLASS POF keyword 222
MAP_DATACLASS_ALT POF keyword 222
MAP_EXPDT POF keyword 222
MAP_MGMTCLASS POF keyword 223
MAP_MGMTCLASS_ALT POF keyword 223
MAP_PREFIX POF keyword 223
MAP_PRIQTY POF keyword 223
MAP_RETPD POF keyword 223
MAP_SECQTY POF keyword 223
MAP_STORAGECLASS POF keyword 223
MAP_STORAGECLASS_ALT POF keyword 223
MAP_THRESHOLD POF keyword 223
MAP_UNIT POF keyword 224
MAP_UNIT_ALT POF keyword 224
mark baseline recoverable worklist command 264
MAX command 30
MAX_CYL POF keyword 224
MAX_FAILURE attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
MAX_PRIQTY POF keyword 224
MAX_SECQTY POF keyword 224
MAX_UNITCNT POF keyword 224
MAXINITS AEXPIN keyword 63
MAXPARTITIONS attribute 614, 615
MAXPARTS attribute 45
MAXRULES attribute 614, 615
MAXSYSREC ALUIN keyword
  description 96
  using multitasking with BMC LOADPLUS 284
  using multitasking with BMC UNLOAD PLUS 276
MAXSYSUT ALUIN keyword 96, 142
MAXSYSUT installation option 142, 171
MAXVALUE attribute 612
MAXVER ALUIN keyword 97
MEMBER symbolic variable 150
MEMBR symbolic variable 150
MEMLIMIT POF keyword 224
MERGECOPY worklist command 359
MGMTCLAS attribute 613
MGMTCLAS installation option 142, 171
MGMTCLASS attribute 589, 590
MI symbolic variable 151
MI TEMPLATE descriptor variable 151
MIGALL (migrate all) command 30
MIGAU attribute
  modifying for a routine in a REPLICATE WORKID statement 594
MIGLOCATIONS ALUIN keyword 97
MIGPROFILE ALUIN keyword 77, 98
MIGRATE command 30
MIGRATE statement 568
MIGRATE TABLES statement, using TCPART 597
migrate worklist command 327
migrating
databases 568
dependent objects 568
objects 557
storage groups 568
table spaces with JOIN keyword 602
tables 568
tables with ROWID columns, using BMC
  BASIC UNLOAD 269
tables with ROWID columns, using BMC
  LOADPLUS 287
tables with ROWID columns, using BMC
  UNLOAD PLUS 278
tables with ROWID columns, using IBM LOAD
  323
tables with ROWID columns, using IBM
  UNLOAD PLUS 346
migrating image copy data 302
MIGSCOPE ALUIN keyword 98
MININITS AEXPIN keyword 63
MINUTE OUTPUT descriptor variable 151
MINUTE symbolic variable 151
MINUTE TEMPLATE descriptor variable 151
MINVALUE attribute 612
MM (block move) action code 32
MMDD symbolic variable 151
MO (migrate options) action code 32
MO symbolic variable 151
MO TEMPLATE descriptor variable 151
MODAUTH attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
MODEDATE attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
MODIFY STATISTICS worklist command 359
modifying for an index partition in a REPLICATE WORKID statement 589, 590
MODTIME attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
MONTH OUTPUT descriptor variable 151
MONTH symbolic variable 151
MONTH TEMPLATE descriptor variable 151
MONTH.DAY OUTPUT descriptor variable 151
MONTH.DAY TEMPLATE descriptor variable 151
MOREVOLS attribute 589, 590
MP (materialized query table parameters) command 30
MQT (materialized query table) command 30
MSSID symbolic variable 151, 315
MT (materialized query text) command 30
MT (MQT text) action code 32
multitasking, more than 256 partitions
  BMC LOADPLUS 284
  BMC UNLOAD PLUS 276
  IBM UNLOAD 346
MULTITBDS ALUIN keyword 98
NAME attribute
  change rules for application objects 50
  change rules for DB2 objects 45
  for SET and WHERE clauses in a column 605, 606
  for SET and WHERE clauses in a database 606, 607
  for SET and WHERE clauses in a field 606, 607
  for SET and WHERE clauses in a parm 609, 610
  for SET and WHERE clauses in a routine 610–612
  for SET and WHERE clauses in a sequence 612
  for SET and WHERE clauses in a storage group 613
  for SET and WHERE clauses in a synonym 613
  for SET and WHERE clauses in a table 613, 614
  for SET and WHERE clauses in a table space 614, 615
  for SET and WHERE clauses in a trigger 615, 616
  for SET and WHERE clauses in a view 616
  for SET and WHERE clauses in an alias 604, 605
  for SET and WHERE clauses in an index 607, 608
native stored procedures worklist command 341
NEW_ACTIVE attribute
  modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_ALNAME attribute 575
NEW_ALOCUNIT attribute 589, 590
NEW_ALOWNER attribute 575
NEW_ASUTIME attribute
  modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_AUDITING attribute 586, 587
NEW_AUTHIDSW attribute 590, 591
NEW_BPOOL attribute 577–579, 588, 589
NEW_CCSID attribute 577, 586–589
NEW_CHECK attribute 590, 591
NEW_CKNAME attribute 576
NEW_CLNAME attribute 576, 577, 579
NEW_CLUST attribute 578, 579
NEW_COLLID attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_COLL attribute 576, 577
NEW_COLNAME attribute 580, 581
NEW_COLNO attribute 576, 577, 579
NEW_COLSSCALE attribute 576, 577
NEW_COLSEQ attribute 580, 581
NEW_COLSNTXT attribute 590, 591
NEW_COLTYPE attribute 576, 577
NEW_COMMIT_ON_RETURN attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_COMPRESS attribute 589, 590
NEW_CONCUR_ACC_RES attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_COPY attribute 578, 579
NEW_CRULE attribute 578, 579, 588, 589
NEW_DATACAPTURE attribute 586, 587
NEW_DATACLASS attribute 589, 590
NEW_DATE_FORMAT attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_DBINFO attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_DBNAME attribute
modifying for a database in a REPLICATE
WORKID statement 577
modifying for a table space in a REPLICATE
WORKID statement 588, 589
table column 586, 587

NEW_DBOWNER attribute 577
NEW_DEBUG_MODE attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_DEC31 attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_DEC31_SCALE attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_DEFAULTVALUE attribute 576, 577
NEW_DEFERPREP attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585
NEW_DEFERPREPARE attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_DEFTI attribute 576, 577
NEW_DEGREE attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_DELETERULE attribute 581
NEW_DETERMINISTIC attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_DEVTYPE attribute 589, 590
NEW_DROPRESTRICT attribute 586, 587
NEW_DSSIZE attribute 588, 589
NEW_DTNAME attribute 576, 577
NEW_DTSHEMA attribute 576, 577
NEW_DYNAMICRULES attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_ENCODING_CCSID attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_EPROC attribute 586, 587
NEW_ERULE attribute 578, 579, 588, 589
NEW_EXPLAIN attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEWEXTERNAL_NAME attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_EXTERNAL_SECURITY attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_FKNAME attribute 581
NEW_FOR_UPDATE_CLAUSE attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_FBOWN attribute 577
NEW_FBROPROC attribute 576, 577
NEW_FREEPAGE attribute 589, 590
NEW_GBPCACHE attribute 589, 590
NEW_IMMEDWRITE attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585

NEW_JAR_ID attribute

NEW_JAR_ID attribute 578, 579
NEW_JXOWNER attribute 578, 579
NEW_JAR_ID attribute
modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_JARSCHEMA attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_JAVA_SIGNATURE attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_KEEPDYNAMIC attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_LABEL attribute modifying for a column in a REPLICATE WORKID statement 576, 577
modifying for a table in a REPLICATE WORKID statement 586, 587
modifying for a view in a REPLICATE WORKID statement 590, 591
NEW_LANGUAGE attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_LENGTH attribute modifying for a parm column in a REPLICATE WORKID statement 579, 580
NEW_LOCATOR attribute modifying for a parm column in a REPLICATE WORKID statement 579, 580
NEW_LOCKMAX attribute 588, 589
NEW_LRULE attribute 588, 589
NEW_MAX_FAILURE attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_MAXROWS attribute 588, 589
NEW_MEMCLUST attribute 588, 589
NEW_MGMTCLASS attribute 589, 590
NEW_MOREVOLS attribute 589, 590
NEW_NAME attribute 576, 577
modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_NUML attribute 576, 577
NEW_OBID attribute 586, 587
NEW_OPTHINT attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_ORDERING attribute 579
NEW_ORDINAL attribute modifying for a parm column in a REPLICATE WORKID statement 579, 580
NEW_ORIGIN attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_OWNER attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_PACKAGEPATH attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_PARAMETER_CCSID attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_PARAMETER_STYLE attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_PARAMETER_VARCHARFORM attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_PARMNAME attribute modifying for a parm column in a REPLICATE WORKID statement 579, 580
NEW_PARTID attribute 589, 590
NEW_PARTS attribute 578, 579, 588, 589
NEW_PATHSCHEMAS attribute 590, 591
modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_PCTFREE attribute 589, 590
NEW_PGSIZE attribute 578, 579
NEW_PIECESIZE attribute 578, 579
NEW_PQTY attribute 589, 590
NEW_PRIKEYNO attribute 576, 577
NEW_PROGRAM_TYPE attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_QUALIFIER attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_QUALVIEW attribute 590, 591
NEW_REF_TBNAME attribute 586, 587
NEW_REF_TBOWNER attribute 586, 587
NEW_REFTBNAME attribute 581
NEW_REFTBOWNER attribute 581
NEW_RELEASE attribute modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_REMS attribute
modifying for a column in a REPLICATE WORKID statement 576, 577
modifying for a routine in a REPLICATE WORKID statement 581–585
modifying for a table in a REPLICATE WORKID statement 586, 587
modifying for a view in a REPLICATE WORKID statement 590, 591
NEW_REOPT attribute
modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_RESULT_SETS attribute
modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_ROUNDING attribute
modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_ROWTYPE attribute
modifying for a parm column in a REPLICATE WORKID statement 579, 580
NEW_RUNOPTS attribute
modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_SCALE attribute
modifying for a parm column in a REPLICATE WORKID statement 579, 580
NEW_SCHEMA attribute
modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_SECSIZE attribute 588, 589
NEW_SYNAME attribute
modifying for a synonym in a REPLICATE WORKID statement 586
NEW_TFINAME attribute
modifying for a table in a REPLICATE WORKID statement 586
NEW_TDNAME attribute
modifying for an alias in a REPLICATE WORKID statement 575
NEW_TBOWNER attribute
modifying for a synonym in a REPLICATE WORKID statement 586
NEW_TBNAME attribute
modifying for a table in a REPLICATE WORKID statement 586
NEW_TBOWNER attribute
modifying for a synonym in a REPLICATE WORKID statement 586
NEW_TBNAME attribute
modifying for a table in a REPLICATE WORKID statement 586
NEW_TIME_FORMAT attribute
modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_TNAME attribute 586–589
NEW_TOWNER attribute 588, 589
NEW_TSTYPE attribute 588, 589
NEW_TYPENAME attribute
modifying for a parm column in a REPLICATE WORKID statement 579, 580
NEW_TYPESCHEMA attribute
modifying for a parm column in a REPLICATE WORKID statement 579, 580
NEW_URULE attribute 578, 579
NEW_VALIDATE attribute
modifying for a routine in a REPLICATE WORKID statement 581–585
NEW_VCATNAME attribute
modifying for a synonym in a REPLICATE WORKID statement 586
NEW_VCATPASS attribute 585, 589, 590
NEW_VPROC attribute 586, 587
NEW_VWNAME attribute 590, 591
NOREGENIDENTITY ALUIN keyword 106, 122, 292
NOREMARKS ALUIN keyword 45, 106
NOREORGONLINE ALUIN keyword 106, 143
NOREPLACE ALUIN keyword 107, 124
NOREPLACE CM/PILOT keyword 355
NOROLEOWN ALUIN keyword 107
NOSEGCSIZE ALUIN keyword 45, 107
NOSPACTIVE parameter, OVERRIDE ALUIN keyword 114
NOSPANNED ALUIN keyword 107
NOSPASUTIME parameter, OVERRIDE ALUIN keyword 114
NOSPCCSID parameter, OVERRIDE ALUIN keyword 114
NOSPCCSID parameter, OVERRIDE ALUIN keyword 114
NOSPCLLOCCID parameter, OVERRIDE ALUIN keyword 114
NOSPDEGREE parameter, OVERRIDE ALUIN keyword 114
NOSPDETERMINISTIC parameter, OVERRIDE ALUIN keyword 115
NOSPDEFERPREPARE parameter, OVERRIDE ALUIN keyword 115
NOSPDECIMAL parameter, OVERRIDE ALUIN keyword 114
NOSPDEXPLAIN parameter, OVERRIDE ALUIN keyword 115
NOSPEXTNAME parameter, OVERRIDE ALUIN keyword 115
NOSPIMMEDWRITE parameter, OVERRIDE ALUIN keyword 115
NOSPISOLEVEL parameter, OVERRIDE ALUIN keyword 115
NOSPJAVA parameter, OVERRIDE ALUIN keyword 115
NOSPKEEPDYN parameter, OVERRIDE ALUIN keyword 115
NOSPLANGUAGE parameter, OVERRIDE ALUIN keyword 115
NOSPMAXFAIL parameter, OVERRIDE ALUIN keyword 115
NOSPOTHINT parameter, OVERRIDE ALUIN keyword 116
NOSPPACKOWNER parameter, OVERRIDE ALUIN keyword 116
NOSPPARAMETERs parameter, OVERRIDE ALUIN keyword 116
NOSPARMCCSID parameter, OVERRIDE ALUIN keyword 116
NOSPARMDATATYPE parameter, OVERRIDE ALUIN keyword 116
NOSPARMLOC parameter, OVERRIDE ALUIN keyword 116
NOSPARMNAME parameter, OVERRIDE ALUIN keyword 116
NOSPARMORDER parameter, OVERRIDE ALUIN keyword 116
NOSPARMSTYLE parameter, OVERRIDE ALUIN keyword 116
NOSPARMVCHAR parameter, OVERRIDE ALUIN keyword 116
NOSPREDRIVER parameter, OVERRIDE ALUIN keyword 116
NOSPRESULTSET parameter, OVERRIDE ALUIN keyword 117
NOSPQUALIFIER parameter, OVERRIDE ALUIN keyword 117
NOSPREGISTERS parameter, OVERRIDE ALUIN keyword 117
NOSPRELEASEAT parameter, OVERRIDE ALUIN keyword 117
NOSPRESULTSET parameter, OVERRIDE ALUIN keyword 117
NOSPREDRIVER parameter, OVERRIDE ALUIN keyword 117
NOSPRESULTSET parameter, OVERRIDE ALUIN keyword 117
NOSPREDRIVER parameter, OVERRIDE ALUIN keyword 117
NOSPREDRIVER parameter, OVERRIDE ALUIN keyword 117
NOSPREDRIVER parameter, OVERRIDE ALUIN keyword 117
NOSPSECURITY parameter, OVERRIDE ALUIN keyword 117
NOSPSQLPATH parameter, OVERRIDE ALUIN keyword 117
NOSPSTAYRESIDENT parameter, OVERRIDE ALUIN keyword 117
NOSPTEXT parameter, OVERRIDE ALUIN keyword 117
NOSPTFORMAT parameter, OVERRIDE ALUIN keyword 117
NOSPVALIDATE parameter, OVERRIDE ALUIN keyword 117
NOSPWLMNEST parameter, OVERRIDE ALUIN keyword 118
NOSQLCOMP AEXIN keyword 59
NOSTARTOVER AEXIN keyword 59
NOSTARTVALUE ALUIN keyword 45, 107
NOTABLE ALUIN keyword 108, 144
NOTABLEACCESS ALUIN keyword 108, 144
NOTABLESPACE ALUIN keyword 45, 108
NOTBAUDIT parameter, OVERRIDE ALUIN keyword 118
NOTBCOLORDER parameter, OVERRIDE ALUIN keyword 118
NOTBDEFVALUE parameter, OVERRIDE ALUIN keyword 118
NOTBTSAUTO parameter, OVERRIDE ALUIN keyword 118
NOTCPROTATE ALUIN keyword 109, 133
NOTIFY AJXIN keyword 67
NOTIFYUNLD AEXIN keyword 60
NOTTRACKMOD ALUIN keyword 45, 109
NOTRIGGER ALUIN keyword 109
NOTSCLOSE parameter
  OVERRIDE ALUIN keyword 118
NOUNIQUECNST ALUIN keyword 109
NOUNLDCOLL ALUIN keyword 109, 144
NOUNLOADEMPTY ALUIN keyword 109, 145
NOUSENGTR ALUIN keyword 110, 145
NOTULICOPY ALUIN keyword 110, 145
NOVALIDATE ALUIN keyword 110
NOVIEW ALUIN keyword 110
NOVIEWCHECK ALUIN keyword 45, 110
NOVALPROP ALUIN keyword 110, 145
NOVWQUAL parameter, OVERRIDE ALUIN keyword 118
NOWKIDREPLACE AEXIN keyword 60
NOWKIDREPLACE keyword 350
NOWLORDER ALUIN keyword 110
NOXMLDATA ALUIN keyword 110
NULL_CALL attribute
  for SET and WHERE clauses in a routine 610–612
NULLS attribute 605, 606
NULLTYPE T1 NULLCHAR ? control card 275
number variable 377
NUPARTS attribute 45
NUPARTS parameter 315

O

O (options) action code 32
OBID attribute
  for SET and WHERE clauses in a table 613, 614
  for SET and WHERE clauses in a trigger 615, 616
  modifying for a table in a REPLICATE WORKID statement 586, 587
OBJECTS ALUIN keyword 110
objects, performable actions using DML 557
OBJINFO ALUIN keyword 111
OBJT symbolic variable 151
OBJTYP symbolic variable 151
OBNAM symbolic variable 151
OBNO symbolic variable 151
omitting objects with Unicode attributes, SET OmitUnicode statement 592
OPTHINT attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
OPTION parameter 301
OPTIONS option, EXPORT command 271
options, -OPTS CM/PILOT worklist command 355
ORDER attribute 605, 606, 612
ORDERBY ALUIN keyword 104, 111
ORDERING attribute 579, 608
ORDINAL attribute
  for SET and WHERE clauses in a parm 609, 610
  modifying for a parm column in a REPLICATE WORKID statement 579, 580
ORIGIN attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585

ORTPARM_DSN POF keyword 225
OUTCOPY option, IMPORT command 302
OUTPUT descriptor
  used in -BMCC worklist command 270
  used in -BMCD worklist command 274
  used in -BMCV worklist command 300
  used with BMC UNLOAD PLUS utility 279
  used with NGT Copy utility 270
  used with NGT Recover utility 270
OUTPUT descriptor variables, list of 145
OVERRIDE ALUIN keyword 112
OVERRIDE(ENQFAILRC4) parameter, OVERRIDE ALUIN keyword 112
OWNER attribute
  change rules for application objects 50
  change rules for DB2 objects 45
  for SET and WHERE clauses in a parm 609, 610
  for SET and WHERE clauses in a routine 610–612
  for SET and WHERE clauses in a trigger 615, 616
  modifying for a routine in a REPLICATE WORKID statement 581–585

P

P (protect) action code 32
PA (path) command 31
PA (propagate attributes) action code 32
PA symbolic variable 151
PA TEMPLATE descriptor variable 151
PACKAGEPATH attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
packages, profile change rules 45
PADDED attribute 607, 608
PARALLEL AEXPIN keyword 63, 119
PARALLEL ALUIN keyword 119, 143
PARALLEL installation option 143, 171
PARAMETER_CCSID attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
PARAMETER_STYLE attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
PARAMETER_VARCHARFORM attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
PARM DML keyword 559
PARMNAME attribute
  for SET and WHERE clauses in a parm 609, 610
  modifying for a parm column in a REPLICATE WORKID statement 579, 580
parms
  attributes for SET and WHERE clauses 609, 610
  DML keyword 559
  modifying attributes in a REPLICATE WORKID statement 579, 580
performable actions 557
PART OUTPUT descriptor variable
  BMCCP symbolic variable 146
  PA symbolic variable 151
PART symbolic variable 151
PART4 symbolic variable 151
PART5 symbolic variable 151
UP symbolic variable 155
UPART symbolic variable 155
PART symbolic variable 151
PART TEMPLATE descriptor variable
  BMCCP symbolic variable 146
  DSNUM symbolic variable 147
  LPART symbolic variable 150
  PART symbolic variable 151
  PART4 symbolic variable 151
  PART5 symbolic variable 151
  UP symbolic variable 155
  UPART symbolic variable 155
PART4 symbolic variable 151
PART5 symbolic variable 151
PARTCOPY ALUIN keyword 120, 143
PARTCPY installation option 171
NOPARTCOPY ALUIN keyword 104
overriding keywords 143
PARTCOPY ALUIN keyword 120
PARTID attribute 589, 590
PARTITION attribute 608, 609, 615
partition-level image copies
using -BMCC worklist command 271
using -BMCL worklist command 283
using -BMCR worklist command 295
using -JCLP worklist command 315
using -SPAC worklist command 337

partitionNumber variable 377
PARTS attribute 578, 579, 588, 589
PARTTYPE attribute for SET and WHERE clauses in a routine 610–612
PATH attribute for SET and WHERE clauses in a table space partition 615
PATHSCHEMAS attribute for SET and WHERE clauses in an index partition 608, 609
modifying for a routine in a REPLICATE WORKID statement 589, 590
modifying for an index partition in a REPLICATE WORKID statement 589, 590

PE (propagate estimates) action code 32
PEALL (propagate estimates) command 31
performance, improving 602
PERIOD attribute 605, 606, 608
PGMR symbolic variable 152
PGSIZE attribute change rules for application objects 50
change rules for DB2 objects 45
for SET and WHERE clauses in a table space partition 615
for SET and WHERE clauses in an index partition 608, 609
modifying for a table space partition in a REPLICATE WORKID statement 589, 590
modifying for an index partition in a REPLICATE WORKID statement 589, 590

PB symbolic variable 152
PB TEMPLATE descriptor variable 152
PCO (parent column) command 31
PCPY1_DATACLASS POF keyword 225
PCPY1_DATACLASS_ALT POF keyword 225
PCPY1_EXPDT POF keyword 225
PCPY1_MGMTCLASS POF keyword 225
PCPY1_MGMTCLASS_ALT POF keyword 225
PCPY1_PREFIX POF keyword 225
PCPY1_PRIQTY POF keyword 225
PCPY1_RETPD POF keyword 225
PCPY1_SECQTY POF keyword 225
PCPY1_STACK POF keyword 225
PCPY1_STORCLASS POF keyword 225
PCPY1_STORCLASS_ALT POF keyword 225
PCPY1_SUPPRESS_SUFF POF keyword 225
PCPY1_THRESH POF keyword 225
PCPY1_UNIT POF keyword 225
PCPY1_UNIT_ALT POF keyword 225

PCPY2_DATACLASS POF keyword 227
PCPY2_DATACLASS_ALT POF keyword 227
PCPY2_EXPDT POF keyword 227
PCPY2_MGMTCLASS POF keyword 227
PCPY2_MGMTCLASS_ALT POF keyword 227
PCPY2_PREFIX POF keyword 227
PCPY2_PRIQTY POF keyword 227
PCPY2_RETPD POF keyword 227
PCPY2_SECQTY POF keyword 227
PCPY2_STACK POF keyword 227
PCPY2_STORCLASS POF keyword 227
PCPY2_STORCLASS_ALT POF keyword 227
PCPY2_SUPPRESS_SUFF POF keyword 227
PCPY2_THRESH POF keyword 227
PCPY2_UNIT POF keyword 227
PCPY2_UNIT_ALT POF keyword 227

PCPY2_UNIT_ALT POF keyword 228
PCTCOMPRESSED attribute 589, 590
PCTFREE attribute change rules for application objects 50
change rules for DB2 objects 45
for SET and WHERE clauses in a table space partition 615
for SET and WHERE clauses in an index partition 608, 609
modifying for a table space partition in a REPLICATE WORKID statement 589, 590
modifying for an index partition in a REPLICATE WORKID statement 589, 590

PIECEattribute change rules for application objects 50
change rules for DB2 objects 45
for SET and WHERE clauses in an index partition 607, 608
rebuilding indexes 299, 329
PK (primary key) action code 32
PK (primary key) command 31
PKPROP ALUIN keyword 120
PLANMGMT installation option 172
PLANMGMBASIC ALUIN keyword 120
PLANMGMTTEXTENDED ALUIN keyword 120
PLANMGT 143
plans, profile change rules 45
PO (procedure options) action code 32
PO (procedure parameters) action code 32
PO (product options file) description 181
keyword descriptions 189
POF(product options file) keywords, list of 181
sample file 181
POFDATE POF keyword 228
POFDS AJXIN keyword 67, 143
POFDS installation option 143, 172, 181
POFDSN AJXIN keyword 67
POP (pop-up messages) command 31
POSTBASELINE AJXIN keyword 68
POSTBDIAG AJXIN keyword 68
POSTBNAME AJXIN keyword 68
POSTBPROFILE AJXIN keyword 68
POSTCBASE AJXIN keyword 68
POSTCCDL AJXIN keyword 68
POSTCDIAG AJXIN keyword 68
POSTCDL AJXIN keyword 68
POSTCOMPARE AJXIN keyword 68
PQTY attribute
for SET and WHERE clauses in a table space partition 615
for SET and WHERE clauses in an index partition 608, 609
modifying for a table space partition in a REPLICATE WORKID statement 589, 590
modifying for an index partition in a REPLICATE WORKID statement 589, 590
PRE_JOBSTEP_INCLUDE POF keyword 229
PREBASELINE AJXIN keyword 68
PREBDIAG AJXIN keyword 68
PREBNAME AJXIN keyword 68
PREBPROFILE AJXIN keyword 68
PRECBASE AJXIN keyword 68
PRECBASE2 AJXIN keyword 68
PRECCDL AJXIN keyword 68
PRECDIAG AJXIN keyword 68
PRECDL AJXIN keyword 68
PRECISION attribute 612
PRECOMPARE AJXIN keyword 68
PREFIX symbolic variable 152
PREVIOUS command 31
PRIBAC symbolic variable 152
PRIBAC TEMPLATE descriptor variable 152
PRIKEYNO attribute 576, 577
PRIMARY attribute 578, 579
PRIPROFILE AJXIN keyword 69
PRIQTY attribute 45, 50
PROC_BMCCHECK_NAME POF keyword 229
PROC_BMCCHECK_STEP POF keyword 229
PROC_BMCCOPY_NAME POF keyword 229
PROC_BMCCOPY_STEP POF keyword 229
PROC_BMCPPRS_NAME POF keyword 229
PROC_BMCPPRS_STEP POF keyword 230
PROC_BMCLOAD_NAME POF keyword 230
PROC_BMCLOAD_STEP POF keyword 230
PROC_BMCRECOVER_NAME POF keyword 230
PROC_BMCRECOVER_STEP POF keyword 230
PROC_BMCREORG_NAME POF keyword 230
PROC_BMCREORG_STEP POF keyword 230
PROC_BMCSTATS_NAME POF keyword 231
PROC_BMCSTATS_STEP POF keyword 231
PROC_BMCSTOP_NAME POF keyword 231
PROC_BMCSTOP_STEP POF keyword 231
PROC_BMCTRIG_NAME POF keyword 232
PROC_BMCTRIG_STEP POF keyword 232
PROC_BMCUNLOAD_NAME POF keyword 232
PROC_BMCUNLOAD_STEP POF keyword 232
PROC_BMCUPRS_NAME POF keyword 232
PROC_BMCUPRS_STEP POF keyword 232
PROC_DSN1COPY_NAME POF keyword 233
PROC_DSN1COPY_STEP POF keyword 233
PROC_DSNUTILB_NAME POF keyword 233
PROC_DSNUTILB_STEP POF keyword 233
PROC_GEN_SET_VAR POF keyword 233
PROC_IDCAMS_NAME POF keyword 233
PROC_IDCAMS_STEP POF keyword 234
PROC_IEFBR14_NAME POF keyword 234
PROC_IEFBR14_STEP POF keyword 234
PROC_TSO_NAME POF keyword 234
PROC_TSO_STEP POF keyword 234
PROC_USER DEFINED POF keyword 235
procedureName variable 377
PRODUCT AJXIN keyword 69, 143
product changes 14
PRODUCT installation option 69, 143, 172
product options file. See POF
product options file parameters. See AJXPOFIN
keywords
profile, copying 32
PROFLOCATION AJXIN keyword 69
PROFNAME AJXIN keyword 69
PROOWNER AJXIN keyword 69
PROGRAM_TYPE attribute
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE WORKID statement 581–585
PROTECT ALUIN keyword 120
PT (partitions) action code 32
PT (partitions) command 31
publications, related 11
PUNCH_DATACLASS 235
PUNCH_EXPDT POF keyword 235
PUNCH_MGMTCLASS POF keyword 235
PUNCH_PREFIX POF keyword 235
PUNCH_PRIQTY POF keyword 235
Q
QE (quick edit) action code 32
QUALIFIER attribute 50
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
QUALVIEW attribute 590, 591
QUIESCE utility worklist command 360

R
R (delete) action code 32
R (repeat) action code 32
R (report) action code 32
RCPY1_DATACLASS POF keyword 236
RCPY1_DATACLASS_ALT POF keyword 236
RCPY1_EXPDT POF keyword 236
RCPY1_MGMTCLASS POF keyword 236
RCPY1_MGMTCLASS_ALT POF keyword 236
RCPY1_PREFIX POF keyword 237
RCPY1_PRIQTY POF keyword 237
RCPY1_RETPD POF keyword 237
RCPY1_SECQTY POF keyword 237
RCPY1_STACK POF keyword 237
RCPY1_STORCLASS POF keyword 237
RCPY1_STORCLASS_ALT POF keyword 237
RCPY1_SUPPRESS_SUFF POF keyword 237
RCPY1_THRESH POF keyword 239
RCPY1_UNIT POF keyword 239
RCPY1_UNIT_ALT POF keyword 239
RCPY2_DATACLASS POF keyword 238
RCPY2_DATACLASS_ALT POF keyword 238
RCPY2_EXPDT POF keyword 238
RCPY2_MGMTCLASS POF keyword 238
RCPY2_MGMTCLASS_ALT POF keyword 238
RCPY2_PREFIX= POF keyword 238
RCPY2_PRIQTY POF keyword 238
RCPY2_RETPD POF keyword 239
RCPY2_SECQTY POF keyword 239
RCPY2_STACK POF keyword 239
RCPY2_STORCLASS POF keyword 239
RCPY2_STORCLASS_ALT POF keyword 239
RCPY2_SUPPRESS_SUFF POF keyword 239
RCPY2_THRESH POF keyword 239
REBIND PACKAGE command 121
rebind worklist command 329
REBINDADVANCED ALUIN keyword 121, 143
REBINDFAIL AEXIN keyword 60
REBINDFAIL POF keyword 240
REBINDNATIVE ALUIN keyword 121, 122, 143
REBINDRC AEXIN keyword 60
REBINDRC POF keyword 240
REBLD installation option 172
  BMCREBUILD ALUIN keyword 76
  IBMREBUILD ALUIN keyword 92
  NOREBUILD ALUIN keyword 105
  overriding keywords 143
REBUILD ALUIN keyword 121
REBUILD utility 329
REBUILD utility worklist command 329
RECEIVE AJXIN keyword 69
RECOVERDOPT AJXIN keyword 69
RECOVERDOPT POF keyword 240
RECOVEROPT AEXIN keyword 60
RECV01 AJXIN keyword 69, 143
RECV02 AJXIN keyword 69, 143
RECVDD01 installation option 69, 172
  COPYDDN ALUIN keyword 79
  overriding keywords 143
  REGISTER ALUIN keyword 122
RECVDD02 installation option 69, 172
  COPYDDN ALUIN keyword 79
  overriding keywords 143
  REGISTER ALUIN keyword 122
RECVSTAK AJXIN keyword 69
REDEFINE NO parameter 292, 299
REF_TBNAME attribute 586, 587
REF_TBOWNER attribute 586, 587
REFRESH ALUIN keyword 121, 143
REFRESH installation option 121, 173
  NOREFRESH ALUIN keyword 105
  overriding keywords 143
REFTBCREATOR attribute 607, 610
REFTBNAME attribute
  for SET and WHERE clauses in a foreign key 607
  for SET and WHERE clauses in a relation 610
  modifying for a relation in a REPLICATE WORKID statement 581
  modifying for a relation key in a REPLICATE WORKID statement 580, 581
REFTBOWNER attribute 580, 581
REGENADVANCED ALUIN keyword 122, 143
REGENATR installation option 143, 173
REGENIDENTITY ALUIN keyword 106, 122
REGENNATIVE ALUIN keyword 121, 122, 143
REGENNSP installation option 121, 122, 143, 173
REGION POF keyword 240
REGISTER ALUIN keyword
COPYDD01 installation option 141
COPYDD02 installation option 141
description 122
RECVDD01 installation option 143
RECVDD02 installation option 143
REGISTER parameter
registering image copies with BMC
LOADPLUS 283
registering image copies with BMC REORG
PLUS 294
registering image copies with NGT Recover 301
related publications 11
relation keys, CD table column 580, 581
relations
attributes for SET and WHERE clauses 610
DML keyword 559
modifying attributes in a REPLICATE
WORKID statement 581
performable actions 557
RELEASE attribute
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE
WORKID statement 581–585
RELKEY DML keyword 559
RELNAME attribute 607, 610
RELS DML keyword 559
REM ALUIN keyword 123
REMARKS attribute
change rules for DB2 objects 45
for SET and WHERE clauses in a column 605, 606
for SET and WHERE clauses in a routine 610–612
for SET and WHERE clauses in a sequence 612
for SET and WHERE clauses in a trigger 615, 616
for SET and WHERE clauses in a view 616
for SET and WHERE clauses in an alias 604, 605
for SET and WHERE clauses in an index 607, 608
REMS attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585
rename object worklist command 333
REOPT attribute
modifying for a routine in a REPLICATE
WORKID statement 581–585
REOPTVAR attribute
for SET and WHERE clauses in a routine 610–612
REORG installation option 174
BMCREORG ALUIN keyword 76
IBMREORG ALUIN keyword 92
overriding keywords 143
REORG PLUS utility, -BMCR worklist command 293
REORG utility 330
REORG utility worklist command 330
REORG_MAPDB POF keyword 240
REORG_MAPTAB POF keyword 240
REORGALL ALUIN keyword 123, 124, 143
REORGALL installation option 174
overriding keywords 143
REORGALL ALUIN keyword 123
REORGPENDONLY ALUIN keyword 124
REORGDOPT AJXIN keyword 69
REORGDOPT POF keyword 241
REORGNONE ALUIN keyword 106, 123, 124, 143
REORGONL installation option 174
NOREORAGE ONLINE ALUIN keyword 106
overriding keywords 143
REORGONLINE ALUIN keyword 123
REORGONLINE ALUIN keyword 123, 143
REORGOPT AEXIN keyword 60
REORG PENDONLY ALUIN keyword 123, 143
REORGREF ALUIN keyword 106
REORGREF installation option 174
overriding keywords 143
REORGREF ALUIN keyword 123
REORGREF ALUIN keyword 123
REPAIR utility worklist command 361
REPL section CM/PILOT worklist command 356
REPLACE ALUIN keyword 124
REPLACE CM/PILOT keyword 355
REPLACETASKID AEXIN keyword 60
REPLACEWORKID AEXIN keyword 60
REPLACEWORKID ALUIN keyword 124
REPLICATE WORKID statement 576
REPLICATE WORKID statement attributes, list of 576, 577
replicating
changes, using JOIN keyword 602
work IDs 573
REPORT ALUIN keyword 124
REPORT NO parameter 292, 297
REPORT utility worklist command 361
REPORTTYPE ALUIN keyword 124
REPT_DATACLASS POF keyword 241
REPT_DATACLASS_ALT POF keyword 241
REPT_EXPDT POF keyword 241
REPT_MGMTCLASS POF keyword 241
REPT_MGMTCLASS_ALT POF keyword 241
REPT_PREFIX POF keyword 241
REPT_PRIQTY POF keyword 242
REPT_RETPD POF keyword 242
REPT_SECQTY POF keyword 242
REPT_STORCLASS POF keyword 242
REPT_STORCLASS_ALT POF keyword 242
REPT_THRESH POF keyword 242
REPT_UNIT POF keyword 242
REPT_UNIT_ALT POF keyword 242
reserved words 376
RESET option, IMPORT command 302
RESIDENT attribute 50
RESMAXIC ALUIN keyword 124
RESTART AEXIN keyword 60
RESTARTPARM AEXIN keyword 60
RESULT_SETS attribute 573
– 612
modifying for a routine in a REPLICATE
WORKID statement 573
– 585
modifying attributes for SET and WHERE clauses 610–612
DML keyword 559
modifying attributes in a REPLICATE
WORKID statement 573
performable actions 559
ROUTINES DML keyword 559
ROWAVG attribute 586, 587
ROWAVGSTATS attribute 586, 587
ROWID columns
migrating with BMC BASIC UNLOAD 269
migrating with BMC LOADPLUS 287
migrating with BMC UNLOAD PLUS 278
migrating with IBM LOAD 323
migrating with IBM UNLOAD 348
ROWMAX attribute 586, 587
ROWSPKEY attribute 589, 590, 608, 609
ROWSPKEYF attribute 589, 590, 608, 609
ROWTYPE attribute
for SET and WHERE clauses in a parm 609, 610
modifying for a parm column in a REPLICATE
WORKID statement 579, 580
RPLUSYNC installation option 175
RPLUSYNCAUTO ALUIN keyword 125
overriding keywords 144
RPLUSYNCREPLACE ALUIN keyword 126
RPLUSYNCREPLACE ALUIN keyword 125, 144
RPLUSYNCREPLACE ALUIN keyword 126, 144
RPTPL installation option 144, 175
RR (block repeat) action code 32
RSEQ# symbolic variable 152
RTYPE symbolic variable 152
rules
DSN1 command 358
RUNOPTS attribute
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE
WORKID statement 581–585
RUNSTATS utility 333
RUNSTATS utility worklist command 333
RUNTIME_HLQ POF keyword 243
RUNTYP symbolic variable 152

S
S (select) action code 32
S (specify) action code 32
S (steps) action code 32
SC symbolic variable 152
SC TEMPLATE descriptor variable 152

Routine, CD table column 581–585
routines
SCALE attribute 605–607
for SET and WHERE clauses in a parm 609, 610
modifying for a parm column in a REPLICATE WORKID statement 579, 580
SCHED_TRIG_CNTM_JOBS POF keyword 243
SCHEMA attribute
change rules for application objects 50
change rules for DB2 objects 45
for SET and WHERE clauses in a parm 609, 610
for SET and WHERE clauses in a routine 610–612
for SET and WHERE clauses in a sequence 612
for SET and WHERE clauses in a trigger 615, 616
modifying for a parm column in a REPLICATE WORKID statement 579, 580
modifying for a routine in a REPLICATE WORKID statement 581–585
schema variable 377
schemaList variable 377
SCOPE1 ALUIN keyword 126
SCOPE2 ALUIN keyword 127
SCOPEPROFILE ALUIN keyword 126, 127
SCOPERULE1 ALUIN keyword 127
SCOPERULE2 ALUIN keyword 127
SCOPETYPE ALUIN keyword 128
scrollable lists, using the FIND command 29
SDSN ALUIN keyword 144
SDSN installation option 144
SDSNE AJXIN keyword 70, 144
SDSNE installation option 144
SEQ OUTPUT descriptor variable 152
SEC OUTPUT descriptor variable 152
SEC symbolic variable 152
SECOND OUTPUT descriptor variable 152
SECOND symbolic variable 152
SECOND TEMPLATE descriptor variable 152
SEQUENCES DML keyword 559
SEQUENCES DML keyword 559
SET CURRENT PATH statement worklist command 335
SET CURRENT SQL ID worklist command 334, 336
SET OMITUNICODE statement 592
SET PRESERVELIMITKEY statement 593
SET SCHEMA statement worklist command 334
SET SPACE ESTIMATION statement 594
SG object type 19
SGCREATOR attribute 616
SGNAME attribute
SENDONLY AJXIN keyword 70
SEQ OUTPUT descriptor variable
DDDOT symbolic variable 147
DDSEQ symbolic variable 147
JDDN symbolic variable 148
RSEQ# symbolic variable 152
SEQ symbolic variable 152
SEQ# symbolic variable 152
SEQ symbolic variable 152
SEQ# symbolic variable 152
SEQ# symbolic variable 152
SEQI installation option 144, 175
SEQNO attribute 615, 616
SEQUENCE attribute
modifying for a column in a REPLICATE WORKID statement 567, 577
modifying for a parm column in a REPLICATE WORKID statement 579, 580
modifying for a relation key in a REPLICATE WORKID statement 580, 581
modifying for a table space partition in a REPLICATE WORKID statement 589, 590
modifying for an index key in a REPLICATE WORKID statement 579
modifying for an index partition in a REPLICATE WORKID statement 589, 590
sequence objects, profile change rules 45
sequenceName variable 377
sequences
attributes for SET and WHERE clauses 612
DML keyword 559
performable actions 557
SEQUENCES DML keyword 559
SET CURRENT PATH statement worklist command 335
SET CURRENT SQL ID worklist command 334, 336
SET OMITUNICODE statement 592
SET PRESERVELIMITKEY statement 593
SET SCHEMA statement worklist command 334
SET SPACE ESTIMATION statement 594
SG object type 19
SGCREATOR attribute 616
SGNAME attribute
for SET and WHERE clauses in a volume 616
modifying for a database in a REPLICATE
  WORKID statement 577
modifying for a storage group in a REPLICATE
  WORKID statement 585
modifying for a table space partition in a
  REPLICATE WORKID statement 589, 590
modifying for an index partition in a
  REPLICATE WORKID statement 589, 590
SGNEW attribute 589, 590
SGOWNER attribute 585
SHOWRULES ALUIN keyword 129
SHRLEVEL CHANGE 275
SHRLEVEL REFERENCE 275
SHRLEVELCHG ALUIN keyword 129
SHRLEVELREF ALUIN keyword 129
SINGLEPHASE ALUIN keyword 129
SLIB variables, list of 145
SMSINCLUDE ALUIN keyword 129
SN symbolic variable 152
SN TEMPLATE descriptor variable 152–154
SORT command 31
SORTDEVT ALUIN keyword 129
SORTNUM AJXIN keyword 70
SORTWK_NBR POF keyword 243
SORTWK_PRIQTY POF keyword 243
SORTWK_SECQTY POF keyword 243
SORTWK_UNIT POF keyword 243
SOURCETYPE ALUIN keyword 130
SP object type 19
SPACE attribute 589, 590, 608, 609
space estimation 594
space estimation source worklist command 337
SPALTER attribute 51
SPANNED installation option 144, 175
SPANNEDRECS 130
SPBXPRINT AEXIN keyword 60
SPE object type 19
SPECIAL_REGS attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE
  WORKID statement 581–585
Specification
  abbreviations for object types 19
  FIND command, using with scrollable lists 29
SPLIT_ROWS attribute 613, 614
SPN object type 19
SPNA object type 19
SPNAME symbolic variable 153
SPNEW attribute
  modifying for a parm column in a REPLICATE
  WORKID statement 579, 580
  modifying for a routine in a REPLICATE
  WORKID statement 581–585
SPOWNER parameter, OVERRIDE ALUIN
  keyword 118
SPP installation option 144, 176
SPS object type 19
SPTEXT ALUIN keyword 130, 141
SPTEXT-TRACE parameter, OVERRIDE ALUIN
  keyword 119
SPTYPE attribute 51
SQ object type 19
SQ symbolic variable 153
SQ TEMPLATE descriptor variable 153
SQL statement worklist command 340
SQL_DATA_ACCESS attribut
  modifying for a routine in a REPLICATE
  WORKID statement 581–585
SQL_DATA_ACCESS attribute
  for SET and WHERE clauses in a routine 610–612
SQL_EXP_LOAD POF keyword 243
SQTY attribute 608, 609
  for SET and WHERE clauses in a table space
  partition 615
  modifying for a table space partition in a
  REPLICATE WORKID statement 589, 590
  modifying for an index partition in a
  REPLICATE WORKID statement 589, 590
SRTOUT_DATACLASS POF keyword 244
SRTOUT_DATACLASS_ALT POF keyword 244
SRTOUT_EXPDT POF keyword 244
SRTOUT_MGMTCLASS POF keyword 244
SRTOUT_MGMTCLASS_ALT POF keyword 244
SRTOUT_PREFIX POF keyword 244
SRTOUT_PRIQTY POF keyword 244
SRTOUT_RETPD POF keyword 244
SRTOUT_SECQTY POF keyword 244
SRTOUT_STORCLASS POF keyword 245
SRTOUT_STORCLASS_ALT POF keyword 245
SRTOUT_THRESH POF keyword 245
SRTOUT_UNIT POF keyword 245
SRTOUT_UNIT_ALT POF keyword 245
SSID AEXIN keyword 61
SSID AJXIN keyword 70
SSID ALUIN keyword 130, 131
SSID CM/PILOT keyword 355
SSID OUTPUT descriptor variable
  GRPNM symbolic variable 147
  JSSID symbolic variable 149
  MSSID symbolic variable 151
  SS symbolic variable 153
  SSID symbolic variable 153
  TSSID symbolic variable 154
SSID symbolic variable 153
SSID TEMPLATE descriptor variable
  GRPNM symbolic variable 147
  JSSID symbolic variable 149
  MSSID symbolic variable 151
  SSID symbolic variable 153
  TSSID symbolic variable 154
SSID symbolic variable 153
SSID TEMPLATE descriptor variable
  GRPNM symbolic variable 147
  JSSID symbolic variable 149
  MSSID symbolic variable 151
  SSID symbolic variable 153
  TSSID symbolic variable 154
ST symbolic variable 153
START attribute 612
STARTOVER AEXIN keyword 61
STARTVALUE attribute 45
statementNumber variable 377
STATHIST installation option 176
  HISTORYALL ALUIN keyword 92
  NOHISTORY ALUIN keyword 102
overriding keywords 144
STATS AEXIN keyword 61
STATS installation option 136, 144, 176
STATUS attribute
  DELETE SYNTABLE statement 563
  DELETE WORKID statement 565
STATUSINFO AEXPIN keyword 63
STAYRESIDENT attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE
    WORKID statement 581–585
STEP_INCLUDE_MEMBER POF keyword 245
STEP# symbolic variable 153
STEPN symbolic variable 153
STEPNAME OUTPUT descriptor variable
  STEPN symbolic variable 153
  TU1 symbolic variable 154
  TU2 symbolic variable 154
  TU3 symbolic variable 154
  UDOPT symbolic variable 154
  WKOWN symbolic variable 156
  WKOWNER symbolic variable 156
STEPNAME symbolic variable 153
STEPNAME TEMPLATE descriptor variable
  STEPN symbolic variable 153
  TU1 symbolic variable 154
  TU2 symbolic variable 154
  TU3 symbolic variable 154
  UDOPT symbolic variable 154
  WKOWN symbolic variable 156
  WKOWNER symbolic variable 156
STGROUP attribute 606
STOG_VCAT attribute 45, 51
STROGROUP attribute 45
STROGROUPS DML keyword 559
stop worklist command 343
STOPCOMM installation option 107, 144, 176
STOPCOMMIT ALUIN keyword 131, 144
STOPCOMMIT installation option 131
STOPICP ALUIN keyword 72
See also ALLOWICP ALUIN keyword
STOPICP installation option 72, 131, 141, 144
See also DUAL
STOPLIST ALUIN keyword 132, 144
STOPLIST installation option 108, 132, 144, 176
STOPWAIT AEXIN keyword 61
STOPWAIT POF keyword 245
STOPWTSECS AEXIN keyword 62
STOPWTSECS POF keyword 245
storage groups
  attributes for SET and WHERE clauses 613
  CD table column 585
  DML keyword 559
  performable actions 557
  profile change rules 45
storageGroupName variable 377
STORCLAS attribute 613
STORCLAS installation option 144, 177
STORCLASS attribute 589, 590
stored procedures, profile change rules 45
STORNAME attribute 608, 609, 615
STOSPACE utility worklist command 362
STYPE attribute 589, 590
SUBEC attribute
  modifying for a routine in a REPLICATE
    WORKID statement 581–585
SUBET attribute
  modifying for a routine in a REPLICATE
    WORKID statement 581–585
SUBMIT AJXIN keyword 70
subselect variable 377
subsystemID variable 377
SUBTYPE attribute 576, 577
for SET and WHERE clauses in a parm 609, 610
modifying for a parm column in a REPLICATE
WORKID statement 579, 580
summary of changes 14
SUPPRESS_COMMENTS POF keyword 246
suppress-type change rule 45
SY (synonym) action code 32
SY object type 19
SYLBnn OUTPUT descriptor 279
SYLBnn TEMPLATE descriptor 346
symbolic variables, list of 145
SYNAME attribute 586
SYNC option, IMPORT command 302
csync point worklist command 343
csync table entries, deleting 563
SYNDELETE AEXIN keyword 62
SYNDELETE POF keyword 246
SYNCPNT installation option 132, 144, 177
SYNCPNT ALUIN keyword 132, 144
SYNFLIST AEXIN keyword 62
synonymName variable 377
synonymOwner variable 377
synonyms
  attributes for SET and WHERE clauses 613
  DML keyword 559
  modifying attributes in a REPLICATE
  WORKID statement 586
  performable actions 557
  profile change rules 45
SYNONYMS DML keyword 559
syntax diagrams
  for CDL statements 377
  for DML statements 558
syntax statement conventions 12
SYOWNER attribute 586
SYSCSTAK AJXIN keyword 70
SYSDISC YES option of DELETEFILES NO
  parameter 288
SYSEXEC POF keyword 246
SYSLIB symbolic variable 153
SYSTEM_MLIB POF keyword 246
SYSTIMESENSITIVE attribute
  for SET and WHERE clauses in a routine 610–612
SYSTYPE installation option 144, 177
SYSUID symbolic variable 153
SYSUT data sets
  used in the -BMCL command 282
  used in the -BMCR command 293
  used in the -JCLP command 316
SYSUT_DATACLASS POF keyword 246
SYST_DATACLASS_ALT POF keyword 246
SYST_EXPDT POF keyword 246
SYST_MGMTCLASS_ALT POF keyword 246
SYST_PREFIX POF keyword 247
SYST_PRIQTY POF keyword 247
SYST_RETPD POF keyword 247
SYST_SECQTY POF keyword 247
SYST_STORCLASS POF keyword 247
SYST_STORCLASS_ALT POF keyword 247
SYST_THRESH POF keyword 247
SYST_UNIT POF keyword 247
SYST_UNIT_ALT POF keyword 247
SYXCNn OUTPUT descriptor 279
SYXCNn TEMPLATE descriptor 346
SZDEVT AJXIN keyword 70
SZDEVT installation option 70
SZDEVT POF keyword 248

T

table space partitions
  attributes for SET and WHERE clauses 615
  DML keyword 559
  modifying attributes in a REPLICATE
  WORKID statement 589, 590
  performable actions 557
table spaces
  attributes for SET and WHERE clauses 614, 615
  DML keyword 559
  implicitly created 272, 302, 333
  modifying attributes in a REPLICATE
  WORKID statement 588, 589
  performable actions 557
  profile change rules 45
table-controlled partitioning, converting to 597
TABLEACC installation option 177
  overriding keywords 144
TABLEACCESS ALUIN keyword 132, 144
TABLEALL ALUIN keyword 133, 144
TABLEALL installation option 177
  NOTABLEALL ALUIN keyword 108
  overriding keywords 144
  TABLEALL ALUIN keyword 133
tableName variable 377
tableOwner variable 377
TABLEPARTS DML keyword 559
tables
attributes for SET and WHERE clauses 613, 614
DML keyword 559
modifying attributes in a REPLICATE
WORKID statement 588, 589
performable actions 557
profile change rules 45
TABLES DML keyword 559
TABLESPACE attribute 45, 51
tableSpaceName variable 377
TABLESPACES DML keyword 559
TAPE_EXPDT POF keyword 248
TAPE_RETPD POF keyword 248
TAPE_VOLCNT POF keyword 248
TAPE1 POF keyword 248
TAPE2 POF keyword 248
TAPE3 POF keyword 248
task ID CM/PILOT worklist command 357
TASKID AEXIN keyword 62
TB (tables) action code 32
TB object type 19
TBB object type 19
TBBH object type 19
TBBR object type 24
TBBT object type 19
TBC object type 19
TBCR symbolic variable 153
TBCRE symbolic variable 153
TBCREATOR attribute
  for SET and WHERE clauses in a column 605, 606
  for SET and WHERE clauses in a field 606, 607
  for SET and WHERE clauses in a synonym 613
  for SET and WHERE clauses in an alias 604, 605
  for SET and WHERE clauses in an index 607, 608
TBGT object type 19
TBH object type 19
TBLOC attribute 575
TBNAM symbolic variable 153
TBNAME attribute
for SET and WHERE clauses for a field 606, 607
for SET and WHERE clauses for an alias 604, 605
for SET and WHERE clauses in a check constraint 605
for SET and WHERE clauses in a column 605, 606
for SET and WHERE clauses in a foreign key 607
for SET and WHERE clauses in a relation 610
for SET and WHERE clauses in a synonym 613
for SET and WHERE clauses in a trigger 615, 616
for SET and WHERE clauses in an index 607, 608
modifying for a check constraint in a
REPLICATE WORKID statement 576
modifying for a column in a REPLICATE
WORKID statement 576, 577
modifying for a relation in a REPLICATE
WORKID statement 581
modifying for a relation key in a REPLICATE
WORKID statement 580, 581
modifying for a synonym in a REPLICATE
WORKID statement 586
modifying for a table in a REPLICATE
WORKID statement 580, 587
modifying for an alias in a REPLICATE
WORKID statement 575
modifying for an index in a REPLICATE
WORKID statement 578, 579
TBNAM symbolic variable 153
TBNEW attribute 576, 577, 580, 581
TBNODE symbolic variable 153
TBOWNER attribute
for SET and WHERE clauses in a check constraint 605
for SET and WHERE clauses in a trigger 615, 616
modifying for a check constraint in a REPLICATE WORKID statement 576
modifying for a column in a REPLICATE WORKID statement 576, 577
modifying for a relation in a REPLICATE WORKID statement 581
modifying for a relation key in a REPLICATE WORKID statement 576
modifying for an alias in a REPLICATE WORKID statement 575
modifying for an index in a REPLICATE WORKID statement 578, 579
TBQ object type 19
TBQT object type 19
TBR object type 24
TBT object type 19
TBTYPE attribute 586, 587
TCP (table-controlled partitioning) command 31
TCPART attribute 613, 614
TCPART parameter 597
TCPROTATE ALUIN keyword 109, 133
TEMPLATE descriptor
  used in -COPY worklist command 308
  used with IBM UNLOAD utility 346
TEMPLATE descriptor variables, list of 145
TEMPLATE_DSN POF keyword 248
TEMPUNIT POF keyword 248
TERMINATOR POF keyword 248
TIME4 symbolic variable 153
TIMEemplate descriptor variable
  HMS symbolic variable 147
  JHMS symbolic variable 148
  TI symbolic variable 153
  TIME symbolic variable 153
TIME_FORMAT attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE WORKID statement 581–585
TIME4 symbolic variable 153
TIMEParm POF keyword 248
TR (trigger) action code 32
TR object type 19
TRA object type 24
TRAB object type 24
TRACE AEXPIN keyword 63
TRACKMOD attribute 45, 51, 615
TRANSFORM ALUIN keyword 134
TRB object type 19
TRIAL ALUIN keyword 134
TRIAL CM/PILOT keyword 352, 355
TRigEVENT attribute 615, 616
triggerName variable 377
triggerOwner variable 377
triggers
  attributes for SET and WHERE clauses 615, 616
  DML keyword 559
  performable actions 557
  profile change rules 45
TRIGGERS DML keyword 559
triggerText variable 377
TRIGTIME attribute 615, 616
TRTCH POF keyword 249
TS (table spaces) action code 32
TS object type 19
TS OUTPUT descriptor variable
  IS symbolic variable 148
  IX symbolic variable 148
  IXNAME symbolic variable 148
  IXSPC symbolic variable 148
  SN symbolic variable 152
  SPNAME symbolic variable 153
  TBNAM symbolic variable 153
  TS symbolic variable 154
  TSIX symbolic variable 154
  TSNAMe symbolic variable 153, 154
TS symbolic variable 154
TS TEMPLATE descriptor variable
IS symbolic variable 148
TBNAM symbolic variable 153
TBNAME symbolic variable 153
TS symbolic variable 154
TSNAME symbolic variable 154
TSA object type 19
TSCR symbolic variable 154
TSG object type 19
TSI object type 19
TSIX symbolic variable 154
TSNAME attribute
  for SET and WHERE clauses for a table space partition 615
  for SET and WHERE clauses in a table 613, 614
  modifying for a table in a REPLICATE WORKID statement 586, 587
  modifying for a table space in a REPLICATE WORKID statement 588, 589
  modifying for a table space partition in a REPLICATE WORKID statement 589, 590
  modifying for an index partition in a REPLICATE WORKID statement 589, 590
TSNAME symbolic variable 154
TSNEW attribute 589, 590
TSO POFRESET command 31
TSO POFRESET POF command 31
TSOPROGRAM POF keyword 249
TSOSUBEXIT POF keyword 249
TSOWNER attribute 588, 589
TSR object type 25
TSR2 object type 25
TSSID symbolic variable 154, 315
TST object type 19
TU1 symbolic variable 154
TU2 symbolic variable 154
TU3 symbolic variable 154
TVALOFF ALUIN keyword 134, 136
TVALWARN ALUIN keyword 135, 137
TX (text) action code 32
TX (text) command 32
TYPE attribute
DELETE SYNCTABLE statement 563
DELETE WORKID statement 565
for SET and WHERE clauses in a database 606
for SET and WHERE clauses in a table 613, 614
for SET and WHERE clauses in a table space 614, 615
  modifying for a column in a REPLICATE WORKID statement 576, 577
  modifying for a routine in a REPLICATE WORKID statement 581–585
TYPE OUTPUT descriptor variable
  JOBTYP symbolic variable 149
  LOCREM symbolic variable 150
  LR symbolic variable 150
  OBJT symbolic variable 151
  OBJTYP symbolic variable 151
  PB symbolic variable 152
  PRIBAC symbolic variable 152
  RTYPE symbolic variable 152
  RUNTYP symbolic variable 152
  TYPE symbolic variable 154
TYPE symbolic variable 154
TYPENAME attribute 605, 606, 612
  for SET and WHERE clauses in a parm 609, 610
  modifying for a parm column in a REPLICATE WORKID statement 579, 580
TYPESCHEMA attribute 605, 606, 612
  for SET and WHERE clauses in a parm 609, 610
  modifying for a parm column in a REPLICATE WORKID statement 579, 580
U
U (undo) action code 32
U (unprotect) action code 32
UC (unique constraint zoom) action code 32
UC object type 19
UCMD symbolic variable 154
UCNN parameter, OVERRIDE ALUIN keyword 119
UCP object type 19
UCPKPALL installation option 113, 144, 178
UCU object type 19
UDOPT symbolic variable 154
UID symbolic variable 154
ULLQ POF keyword 249
ULLQ symbolic variable 154
UNDOALL (undo all) command 32
  unique constraints, profile change rules 45
uniqueConstraintName variable 377
UNIQUERULE attribute 607, 608
UNLD_FREF_DATACLASS POF keyword 249
UNLD_FREF_DIRBLOCK POF keyword 249
UNLD_FREF_MGMTCLASS POF keyword 250
UNLD_FREF_PREFIX POF keyword 250
UNLD_FREF_PRIQTY POF keyword 250
UNLD_FREF_SECQTY POF keyword 250
UNLD_FREF_STORCLASS POF keyword 250
UNLD_FREF_SUPPR_SUFF POF keyword 250
UNLD_FREF_UNIT POF keyword 250
UNLD1_DATACLASS POF keyword 250
UNLD1_DATACLASS_ALT POF keyword 251
UNLD1_EXPDT POF keyword 251
UNLD1_MGMTCLASS POF keyword 251
UNLD1_MGMTCLASS_ALT POF keyword 251
UNLD1_PREFIX POF keyword 251
UNLD1_PRIQTY POF keyword 251
UNLD1_RETPD POF keyword 251
UNLD1_SECQTY= POF keyword 251
UNLD1_STACK POF keyword 251
UNLD1_STORCLASS POF keyword 251
UNLD1_STORCLASS_ALT POF keyword 251
UNLD1_SUPPRESS_SUFF POF keyword 252
UNLD1_THRESH POF keyword 252
UNLD1_UNIT POF keyword 252
UNLD1_UNIT_ALT POF keyword 252
UNLD2_DATACLASS POF keyword 252
UNLD2_DATACLASS_ALT POF keyword 252
UNLD2_EXPDT POF keyword 253
UNLD2_MGMTCLASS POF keyword 253
UNLD2_MGMTCLASS_ALT POF keyword 253
UNLD2_PRIQTY POF keyword 253
UNLD2_RETPD POF keyword 253
UNLD2_SECQTY POF keyword 253
UNLD2_STACK POF keyword 253
UNLD2_STORCLASS POF keyword 253
UNLD2_STORCLASS_ALT POF keyword 254
UNLD2_SUPPRESS_SUFF POF keyword 254
UNLD2_THRESH POF keyword 254
UNLD2_UNIT POF keyword 254
UNLD2_UNIT_ALT POF keyword 254
UNLD3_DATACLASS POF keyword 254
UNLD3_EXPDT POF keyword 254
UNLD3_MGMTCLASS POF keyword 255
UNLD3_PRIQTY POF keyword 255
UNLD3_RETPD POF keyword 255
UNLD3_STORCLASS POF keyword 255
UNLD3_SUPPRESS_SUFF POF keyword 255
UNLD3_UNIT POF keyword 255
UNLD4_DATACLASS POF keyword 255
UNLD4_MGMTCLASS POF keyword 255
UNLD4_PREFIX POF keyword 256
UNLD4_STORCLASS POF keyword 256
UNLD4_SUPPRESS_SUFF POF keyword 256
UNLD4_UNIT POF keyword 256
UNLD_COLL ALUIN keyword 135, 144
UNLD_COLL installation option 178
NOUNLD_COLL ALUIN keyword 109
overriding keywords 144
UNLDEMPTR installation option 178
NOUNLOADEMPTY ALUIN keyword 109
overriding keywords 145
UNLOADEMPTY ALUIN keyword 135
UNLD_STAK AJXIN keyword 70
UNLOAD PLUS utility
- BMCD worklist command 268, 273
unload record worklist command 348
UNLOAD utility, -UNLI worklist command 345
UNLOADCOPY ALUIN keyword 135
UNLOADCOPYDS ALUIN keyword 135
UNLOADDOPT AEXIN keyword 62
UNLOADDOPT AJXIN keyword 70
UNLOADDOPT POF keyword 256
UNLOADEMPTY ALUIN keyword 135, 145
unloading data 275, 346
UNLOADTABLE ALUIN keyword 136
UP symbolic variable 155
UPART symbolic variable 155
UPDATE statement 597
UPDSTATS installation option 178
ALLSTATSUPD ALUIN keyword 73
BMCSUTSUPD ALUIN keyword 76
DB2STATSUPD ALUIN keyword 80
overriding keywords 145
USE (use) action code 32
USE_NGT_AUTO POF keyword 256
USENGTR ALUIN keyword 136, 145
USENGTR installation option 178
overriding keywords 145
USER_HLQ POF keyword 256
USER_VAR1_CHAR POF keyword 257
USER_VAR2_CHAR POF keyword 257
USER_VAR3_CHAR POF keyword 257
USER_VAR4_CHAR POF keyword 257
USER_VAR5_CHAR POF keyword 257
USER1 symbolic variable 155
USERID OUTPUT descriptor variable
USERID symbolic variable 155
UT symbolic variable 155
UT TEMPLATE descriptor variable
ALID symbolic variable 146
FCMD symbolic variable 147
JQID symbolic variable 149
LI symbolic variable 150
LIST symbolic variable 150
OBJT symbolic variable 151
OBJTYP symbolic variable 151
UCMD symbolic variable 154
USER1 symbolic variable 155
USER2 symbolic variable 155
UT symbolic variable 155
UTIL symbolic variable 155
UTILITYID AEXIN keyword 62
UTILITYID AJXIN keyword 70
ZACCTNUM symbolic variable 157
ZPREFIX symbolic variable 157
ZUSER symbolic variable 157
USERID symbolic variable 155
UVR1 symbolic variable 155
UVR2 symbolic variable 155
UVR3 symbolic variable 155
UVR4 symbolic variable 155
UVR5 symbolic variable 155
V
V (execution status) action code 32
V (view sync table) action code 32
VALIDATE attribute
for SET and WHERE clauses in a routine 610–612
modifying for a routine in a REPLICATE WORKID statement 581–585
VALIDPROC attribute 45, 51
VALOFF ALUIN keyword 136
VALPROC attribute 613, 614
VALUES attribute 45
VALWARN ALUIN keyword 137
variables
  syntax diagram, list of 377
VCAT AEXIN keyword 62
VCAT AJXIN keyword 70
VCAT attribute 45, 51
VCAT symbolic variable 155
VCAT_STOG attribute 45, 51
VCATNAME attribute
  for SET and WHERE clauses in a storage group 613
  for SET and WHERE clauses in a table space partition 615
  for SET and WHERE clauses in an index partition 608, 609
  modifying for a storage group in a REPLICATE WORKID statement 585
  modifying for a table space partition in a REPLICATE WORKID statement 589, 590
  modifying for an index partition in a REPLICATE WORKID statement 589, 590
VCATPASS attribute 585, 589, 590
VERS7FMT AJXIN keyword 70
VERSION attribute 51
  for SET and WHERE clauses in a parm 609, 610
  for SET and WHERE clauses in a routine 610–612
  modifying for a parm column in a REPLICATE WORKID statement 579, 580
  modifying for a routine in a REPLICATE WORKID statement 581–585
VERSIONING_NAME attribute 586, 587
VERSIONING_SCHEMA attribute 586, 587, 613, 614
VERSIONING_TABLE attribute 613, 614
viewName variable 377
viewOwner variable 377
views
  attributes for SET and WHERE clauses 616
  DML keyword 559
  modifying attributes in a REPLICATE WORKID statement 573
  performable actions 557
  profile change rules 45
VIEWS DML keyword 559
VIEWX (view execution) command 32
VL (volume list) action code 32
VL (volume list) command 32
VOLID attribute 616
VOLORDER attribute
  change rules for application objects 51
  change rules for DB2 objects 45
  modifying a table space partition in a REPLICATE WORKID statement 589, 590
  modifying an index partition in a REPLICATE WORKID statement 589, 590
VOLUME attribute 45, 51
volumes
  attributes for SET and WHERE clauses 616
  DML keyword 559
  performable actions 557
VOLUMES DML keyword 559
volumeSerialNumber variable 377
VPROC attribute 586, 587
VRM installation option 145, 179
VT (view text) action code 32
VT (view text) command 26
VVALOFF ALUIN keyword 136, 137
VVALPROP ALUIN keyword 137, 145
VVALPROP installation option 179
  NOVVALPROP ALUIN keyword 110
  overriding keywords 145
  VVALPROP ALUIN keyword 137
VVALWARN ALUIN keyword 137
VW (views) action code 32
VW object type 19
VWNAME attribute 590, 591
VWNEW attribute 590, 591
VWOWNER attribute 590, 591

W
W (edit worklist) action code 32
W (edit worklist) command 32
WARNRC AEXIN keyword 62
WDSN installation option 145
WKID symbolic variable 156
WKNAME attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
WKOWN symbolic variable 156
WKOWNER attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
WKOWNER symbolic variable 156
WLAUTH attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
WLDATE attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
WLM_ENV attribute 51
WLM_ENV_FOR_NESTED attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE
    WORKID statement 581–585
WLM_ENVIRONMENT attribute
  for SET and WHERE clauses in a routine 610–612
  modifying for a routine in a REPLICATE
    WORKID statement 581–585
WLORDER ALUIN keyword 110, 138
WLORDERMSG ALUIN keyword 138
WLPS installation option 145
WLSS installation option 145
WLSTATDATE attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
WLSTATTIME attribute
  DELETE SYNCTABLE statement 563
  DELETE WORKID statement 565
WLM_ENV attribute 51
work ID
  copying 573
  deleting 565
  deleting multiple 565
  replicating 573
WORK_DATACLASS POF keyword 257
WORK_MGMTCLASS POF keyword 257
WORK_STORCLASS POF keyword 257
WORKID AEXIN keyword 62
WORKID AJXIN keyword 71
WORKID ALUIN keyword 138
WORKID symbolic variable 156
WORKID8 symbolic variable 156
WORKIDMERGE ALUIN keyword 139
worklist commands
-AMS 261
-AUTH 262
-BASE 263
-BASF 264
-BEGG 265, 309
-BEGP 265, 310
-BEGU 267, 310
-BIND 267
-BMCC 265, 269
-BMCD 265, 268, 273
-BMCK 265, 280
-BMCL 265, 280
-BMCR 293
-BMCS 265
-BMCU (BMCU Execute a BMC Utility) 357
-BMCSV 298
-CHEK 265, 305
-CMD 307
-DBUG (Debug) 357
-ENDG 265, 309
-ENDP 265, 310
-ENDU 267, 310
-GLID 311
-ISMX 313
-JCLP 314
-LCMD 318
-LDXT 319
-LOAD 321
-MIGR 327
-REBD 329
-REOR 330
-RNAM 333
-RNST 265, 333
-SETS 336
-SPAC 337
-SPBX 338
-SQL 340
-SQLM (SQL Statement) 361
-SQLP 341
-SSID 342
-STOP 343
-SYNC 343
-TIME (file creation time) 345, 375
-UNLI 345
-UNRC (unload record) 348
-WKID 350
-WKID (Work ID) 350
file format 259, 373
processed in parallel 265
worklist, comment lines 261
WORKLISTDDN AJXIN keyword 71
WORKLISTFORMAT AJXIN keyword 71
WPS AJXIN keyword 71
WPS installation option 71
WRK100n data set 293
WSS AJXIN keyword 71
WSS installation option 71

W

XML data
loading with BMC LOADPLUS 287
loading with IBM LOAD 322
unloading with BMC UNLOAD PLUS 279
unloading with IBM UNLOAD 346

XMLCOL parameter
-BMCC worklist command 272
-COPY worklist command 309
-RNST worklist command 333

XMLTBN parameter
-BMCC worklist command 272
-COPY worklist command 309
-RNST worklist command 333

XMLTBO parameter
-BMCC worklist command 272
-COPY worklist command 309
-RNST worklist command 333

XP (XML column index parameters) command 32

Y
YE symbolic variable 156
YE TEMPLATE descriptor variable 156
YEAR OUTPUT descriptor variable 156
YEAR symbolic variable 156
YMD symbolic variable 156
YY OUTPUT descriptor variable 156
YY symbolic variable 156
YYYY symbolic variable 157
YYYYY symbolic variable 157

Z
Z (zoom) action code 32
ZACCTNUM SLIB variable 157
ZACCTNUM symbolic variable 157
ZPARAM installation option 179
ZPREFIX SLIB variable
-PREFIX symbolic variable 152
-ZPREFIX symbolic variable 157
ZPREFIX symbolic variable 157
ZSYSID SLIB variable 157
ZSYSID symbolic variable 157
ZUSER SLIB variable
-UID symbolic variable 154
-USERID symbolic variable 155
-ZUSER symbolic variable 157
-ZUSER symbolic variable 157