



Informatica® PowerExchange
10.4.0

Utilities Guide

Informatica PowerExchange Utilities Guide
10.4.0
December 2019

© Copyright Informatica LLC 2005, 2020

This software and documentation are provided only under a separate license agreement containing restrictions on use and disclosure. No part of this document may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording or otherwise) without prior consent of Informatica LLC.

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation is subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License.

Informatica, the Informatica logo, PowerCenter, and PowerExchange are trademarks or registered trademarks of Informatica LLC in the United States and many jurisdictions throughout the world. A current list of Informatica trademarks is available on the web at <https://www.informatica.com/trademarks.html>. Other company and product names may be trade names or trademarks of their respective owners.

Portions of this software and/or documentation are subject to copyright held by third parties. Required third party notices are included with the product.

The information in this documentation is subject to change without notice. If you find any problems in this documentation, report them to us at infa_documentation@informatica.com.

Informatica products are warranted according to the terms and conditions of the agreements under which they are provided. INFORMATICA PROVIDES THE INFORMATION IN THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT.

Publication Date: 2020-05-05

Table of Contents

Preface	14
Informatica Resources.	14
Informatica Network.	14
Informatica Knowledge Base.	14
Informatica Documentation.	14
Informatica Product Availability Matrices.	15
Informatica Velocity.	15
Informatica Marketplace.	15
Informatica Global Customer Support.	15
 Chapter 1: Introduction to PowerExchange Utilities.....	16
PowerExchange Utilities Overview.	16
PowerExchange Utilities by Operating System.	18
PowerExchange Utilities Syntax Conventions.	19
Environment Variable Incompatibilities Between PowerExchange and PowerCenter.	19
Setting the PowerExchange Environment and Starting a Utility.	20
PowerExchange Sample JCL.	20
 Chapter 2: createdatamaps - Data Map Creation Utility.....	21
createdatamaps Utility Overview.	21
Requirements and Considerations for COBOL Copybooks and DBDs.	22
createdatamaps Command Syntax.	22
Creating, Editing, and Testing a Data Map.	24
Control Files for Data Map Generation.	25
Rules for Control Files.	25
Control File Structure.	26
Schema File for Describing Control Files.	29
Schema File Reference.	31
DatamapGeneration Complex Type.	31
GenBase Complex Type.	31
DataConfigBase Complex Type.	32
SEQGen Complex Type.	32
VSAMGen Complex Type.	33
IMSGen Complex Type.	33
GenConfigBase Complex Type.	34
SEQGenConfig Complex Type.	35
VSAMGenConfig Complex Type.	35
IMSGenConfig Complex Type.	36
DatamapPropertiesBase Complex Type.	36
ParserConfigBase Complex Type.	36

CopybookParserConfig Complex Type.	37
CacheConfig Complex Type.	37
RIDConfig Complex Type.	38
FilePath Complex Type.	39
ImportMetadataBase Complex Type.	40
CopybookImportMetadata Complex Type.	40
DBDImportMetadata Complex Type.	40
OverlayMetadata Complex Type.	40
DatamapInstanceBase Complex Type.	41
SEQDatamapProperties Complex Type.	41
SEQDatamapInstance Complex Type.	42
VSAMDatamapProperties Complex Type.	42
VSAMDatamapInstance Complex Type.	43
IMSDatamapProperties Complex Type.	43
IMSDatamapInstance Complex Type.	44
FilterColumnGroup Complex Type.	45
Filter Complex Type.	45
Log File for Data Maps Creation Utility.	46
JAXB Error Messages.	47
Redefinitions and Record IDs in COBOL Copybooks.	48
How the createdatamaps Utility Creates Records for Redefined Fields and Groups.	48
Use of the maxRedefines Element to Limit the Number of Records.	48
Redefinitions Without Record IDs in COBOL Copybooks.	49
Redefinitions with RID Fields in COBOL Copybooks.	49
Redefine Filler Fields.	54
Copybook and DBD Metadata for IMS Data Maps.	54
Column Name Filters.	55
Unavailable Data Map Properties.	56
Examples.	57
Example: Simple SEQ Data Map.	58
Example: SEQ Data Map with Multiple Records and Tables.	59
Example: Simple VSAM KSDS Data Map.	62
Example: VSAM RRDS Data Maps with Multiple Records and Tables.	63
Example: IMS DBD Import with No COBOL Overlay.	65
Example: IMS DBD Import with COBOL Overlays.	67
Example: Finding RID Fields.	70
Chapter 3: DTLCCADW - Adabas PCAT Utility.....	75
DTLCCADW Utility Overview.	75
DTLCCADW Utility Functions.	75
P (Populate PCAT Control File) Function.	76
R (Report on PCAT Control File) Function.	76
I (Insert) Function.	76

D (Delete) Function.	76
L (Reset Latest Sequence Number) Function.	76
V (Rebuild the PCAT Control File) Function.	76
A (Add) Function.	77
S (Submit ADASEL) Function.	77
T (Submit ET Record Extraction) Function.	77
E (ET/BT Record Extraction) Function.	77
Chapter 4: DTLCUIML - IMS Log Marker Utility.....	78
DTLCUIML Utility Overview.	78
DTLCUIML Utility Parameters.	79
DTLCUIML Utility Reports.	79
SYSPRINT: Control Report.	79
DFSSTAT: IMS Activity Report.	80
User-Defined Log Records.	80
Chapter 5: DTLINFO - Release Information Utility.....	81
DTLINFO Utility Overview.	81
Supported Operating Systems for the DTLINFO Utility.	81
Control Statement Syntax for the DTLINFO Utility.	82
Control Statement Parameters for the DTLINFO Utility.	82
Running the DTLINFO Utility on i5/OS.	82
Running the DTLINFO Utility on Linux, UNIX, and Windows.	82
Running the DTLINFO Utility on z/OS.	83
DTLINFO Utility on i5/OS Examples.	83
DTLINFO Utility on i5/OS - Example 1.	84
DTLINFO Utility on i5/OS - Example 2.	84
DTLINFO Utility on Linux, UNIX, and Windows Examples.	84
DTLINFO Utility on Linux, UNIX, and Windows - Example 1.	84
DTLINFO Utility on Linux, UNIX, and Windows - Example 2.	84
DTLINFO Utility on z/OS Examples.	84
DTLINFO Utility on z/OS - Example 1.	85
DTLINFO Utility on z/OS - Example 2.	85
Chapter 6: DTLREXE - Remote Execution Utility.....	86
DTLREXE Utility Overview.	86
Supported Operating Systems for the DTLREXE Utility.	86
Control Statement Syntax for the DTLREXE Utility.	87
Control Statement Parameters for the DTLREXE Utility.	87
DELETE Statement.	87
PING Statement.	89
SUBMIT Statement.	90
SYSTEM Statement.	92

Running the DTLREXE Utility on i5/OS.	93
Running the DTLREXE Utility on Linux and UNIX.	93
Submitting a Remote z/OS Job Specifying a PDS Member.	93
Submitting a Remote z/OS Job Specifying a Sequential MVS Data Set.	93
Deleting a File from a Remote System.	93
Running a File on a Remote System.	94
Running the DTLREXE Utility on Windows.	94
Running the DTLREXE Utility on z/OS.	94
Running the DTLREXE Utility with PROG=SUBMIT.	94
Running the DTLREXE Utility with PROG=PING.	95
Running the DTLREXE Utility with PROG=DELETE.	95
Running the DTLREXE Utility with PROG=SYSTEM.	96
DTLREXE Utility Usage Notes.	96
DTLREXE Utility on z/OS Example.	97
DTLREXE Utility on z/OS Example JCL.	97
DTLREXE Utility on z/OS Output Data Set.	97
 Chapter 7: DTLUAPPL - Restart Token Utility.....	99
DTLUAPPL Utility Overview.	99
Supported Operating Systems for the DTLUAPPL Utility.	100
DTLUAPPL Control Statement Syntax.	100
Connection Statement.	101
ADD and MOD Statements.	102
END APPL Statement.	105
PRINT APPL Statement.	105
Running the DTLUAPPL Utility on i5/OS.	105
Running the DTLUAPPL Utility on Linux, UNIX, and Windows.	106
Running the DTLUAPPL Utility on z/OS.	106
DTLUAPPL Utility Examples.	108
Example 1. Generating Restart Tokens at the Application Level.	108
Example 2. Generating Restart Tokens at the Capture Registration Level.	108
Example 3. Generating Restart Tokens for Continuous Extraction Mode.	109
Example 4. Adding an Application with Restart Tokens.	109
Example 5. Adding an Application and Generating Restart Tokens on a Remote Instance	110
Example 6. Modifying Restart Tokens in an Application.	110
Example 7. Modifying an Application and Adding a Registration.	110
Example 8. Printing Information for an Application.	110
 Chapter 8: DTLUCBRG - Batch Registration Utility.....	112
DTLUCBRG Utility Overview.	112
Supported Operating Systems for the DTLUCBRG Utility	113
DTLUCBRG Utility Parameters.	113
File Format for the INPUT_FN Option.	121

Expression-Field Functions That Support CREATEBICI Processing	121
DTLUCBRG Utility Source-Specific Parameters.	122
Specifying Multiple Sets of Parameters on the DTLUCBRG Utility.	125
Sample Input for the DTLUCBRG Utility.	125
DTLUCBRG Code Page Processing.	127
Code Page Processing for DB2 Registrations in Local Mode on z/OS.	127
Running the DTLUCBRG Utility.	127
Running the DTLUCBRG Utility on i5/OS.	127
Running the DTLUCBRG Utility on Linux, UNIX, and Windows.	128
Running the DTLUCBRG Utility on z/OS.	128
DTLUCBRG Utility Example Reports.	130
DTLUCBRG Utility with RPTCOLS=N Report Description.	130
DTLUCBRG Utility with RPTCOLS=Y Report Description.	131
DTLUCBRG Utility Usage Notes.	133
Chapter 9: DTLUCDEP - CDEP Maintenance Utility.....	135
DTLUCDEP Utility Overview.	135
Supported Operating Systems for the DTLUCDEP Utility	136
Control Statement Syntax for the DTLUCDEP Utility.	136
Control Statement Parameters for the DTLUCDEP Utility.	136
CDEP Definition Examples.	137
Running the DTLUCDEP Utility on i5/OS.	138
Running the DTLUCDEP Utility on Linux, UNIX, and Windows.	138
Running the DTLUCDEP Utility on z/OS.	138
DTLUCDEP Utility on i5/OS Example.	140
DTLUCDEP Utility on Linux, UNIX, and Windows Example.	140
DTLUCDEP Utility on z/OS Example.	141
Chapter 10: DTLUCSR2 - IDMS SR2 and SR3 Records Utility.....	142
DTLUCSR2 Utility Overview.	142
Running the DTLUCSR2 Utility.	143
Chapter 11: DTLUCUDB - DB2 for Linux, UNIX, and Windows CDC Utility	144
DTLUCUDB Utility Overview.	144
Running the DTLUCUDB Utility.	144
DTLUCUDB Utility Syntax.	144
Command Options for the DTLUCUDB Utility.	146
Gathering Diagnostic Information to Resolve a DB2 Capture Problem.	151
Chapter 12: DTLULCAT and DTLULOGC - IDMS Log Catalog Utilities.....	153
DTLULCAT and DTLULOGC Utilities Overview.	153
Running the DTLULCAT Utility.	154
Running the DTLULOGC Utility.	154

Manually Manipulating the Log Catalog.	156
Guidelines for Adding Logs to the Catalog with the DTLULCAT and DTLULOGC Utilities.	158
Chapter 13: DTLURDMO - Data Map Utility.	160
DTLURDMO Utility Overview.	160
Supported Operating Systems for the DTLURDMO Utility.	161
Control Statement Overview for the DTLURDMO Utility.	161
Control Statement Syntax for the DTLURDMO Utility.	161
Control Statements and Parameters for the DTLURDMO Utility	162
Global Statements.	163
DM_COPY Statement.	169
REG_COPY Statement.	176
XM_COPY Statement.	185
Scope of Operands.	191
Running the DTLURDMO Utility on i5/OS.	193
Running the DTLURDMO Utility on Linux, UNIX, and Windows.	193
Running the DTLURDMO Utility on z/OS.	194
DTLURDMO Control Statement Examples.	194
Copying Selected Data Maps.	195
Copying All Data Maps	195
Copying and Modifying Data Maps	195
Copying Registrations and Generating Extraction Maps	195
Copying Registrations, Generating Extraction Maps, and Merging Extraction Maps with Bulk Data Maps	196
Copying Microsoft SQL Server Registrations and Generating Extraction Maps with a User-Defined Instance ID.	196
Copying IMS Data Maps and Copying and Modifying Registrations.	197
Copying IMS Data Maps and Registrations and Modifying the IMSID Data Map Property.	198
DTLURDMO Report Examples.	199
DM_COPY Test Mode Example.	199
DM_COPY Validate Example.	201
REG_COPY Test Mode Example.	205
REG_COPY Validate Mode Example.	206
Chapter 14: DTLUTSK - Task Control Utility	208
DTLUTSK Utility Overview.	208
DTLUTSK Command Line Utility on i5/OS.	208
DTLUTSK Command Line Utility on Linux, UNIX, and Windows.	210
Displaying DTLUTSK Utility Help.	212
DTLUTSK Job on z/OS.	212
Example JCL for a DTLUTSK Job on z/OS	212
Example Output from a DTLUTSK Job on z/OS.	214
DTLUTSK Command Line Utility on z/OS.	214

LISTTASK Command.	214
STOPTASK Command.	214
LISTLOCATIONS Command.	215
LISTALLOC Command.	215
FREEALLOC Command.	215
Running the DTLUTSK Utility in the PowerExchange Navigator.	216
DTLUTSK Utility Security.	217
DTLUTSK Utility Security Requirements on z/OS.	217
DTLUTSK Utility Security Requirements on i5/OS.	217
Using Signon.txt to Authorize Users to Display or Stop Tasks.	217

Chapter 15: EDMLUCTR - Log Scan and Print Utility..... 218

EDMLUCTR Utility Overview.	218
Supported Operating Systems for the EDMLUCTR Utility.	218
Control Statement Syntax for the EDMLUCTR Utility.	219
Control Statement Parameters for the EDMLUCTR Utility.	219
-SEL Statement.	219
-MASK Statement.	220
Running the EDMLUCTR Utility.	221
EDMLUCTR Utility Usage Notes.	221
EDMLUCTR Utility Examples.	221
EDMLUCTR Utility - Example 1.	222
EDMLUCTR Utility - Example 2.	223
EDMLUCTR Utility - Example 3.	223
EDMLUCTR Utility - Example 4.	224

Chapter 16: EDMXLUTL - Event Marker Utility..... 226

EDMXLUTL Utility Overview.	226
Creating an Event Marker in Batch Mode.	226
EDMXLUTL Utility JCL Statements.	227
EDMXLUTL Utility Control Statements.	227
EDMXLUTL Utility EVENT Command.	227
EVENT Command Syntax.	228
EVENT Command Usage.	228
Keyword Sets for the BASEEDM Category.	228
MARK Keyword Set.	229
NOTIFY Keyword Set.	229
EDMXLUTL Utility Example.	231

Chapter 17: HOSTENT - TCP/IP Address Reporter Utility..... 232

HOSTENT Utility Overview.	232
Supported Operating Systems for the HOSTENT Utility	232
Running the HOSTENT Utility on i5/OS.	233

Running the HOSTENT Utility on Linux and UNIX.	233
Running the HOSTENT Utility on z/OS.	233
HOSTENT Utility Usage Notes.	234
HOSTENT Utility Resolver Details.	234
HOSTENT Utility Output.	235
HOSTENT Utility on i5/OS Example.	235
HOSTENT Utility on Linux and UNIX Example.	236
HOSTENT Utility on z/OS Example.	236
 Chapter 18: PWXCATMY - MySQL Catalog Utility.....	237
PWXCATMY Utility Overview.	237
Supported Operating Systems for the PWXCATMY Utility.	238
PWXCATMY Operation Types.	238
Command Syntax and Parameters.	239
 Chapter 19: PWXUCCLPRT - Print Log Summary Utility.....	241
PWXUCCLPRT Utility Overview.	241
Supported Operating Systems for the PWXUCCLPRT Utility.	241
Command Syntax for the PWXUCCLPRT Utility.	242
PWXUCCLPRT Utility Parameters.	242
PWXUCCLPRT INPUT Statement Parameters.	242
PWXUCCLPRT OUTPUT Statement Parameters.	243
Example PWXUCCLPRT Parameter File	244
Examples of PWXUCCLPRT Output.	244
Example 1. CSV Output Format.	244
Example 2. Dump Output Format.	245
Example 3. Tokens Output Format.	246
 Chapter 20: PWXUCDCT - Logger for Linux, UNIX, and Windows Utility.....	247
PWXUCDCT Utility Overview.	247
Supported Operating Systems for the PWXUCDCT Utility.	248
Control Statement Syntax for PWXUCDCT Commands.	248
PWXUCDCT Commands and Parameters.	248
Commands.	249
Parameter Descriptions.	253
Running the PWXUCDCT Utility.	254
Usage Notes for the PWXUCDCT Utility.	255
Examples of PWXUCDCT Utility Commands.	255
Example 1. Creating a Backup of the CDCT File.	255
Example 2. Restoring the CDCT File from a Backup File.	256
Example 3. Re-creating the CDCT File After a Failure.	256
Example 4. Reporting and Deleting Orphan CDCT Records.	257
Example 5. Reporting and Deleting Expired CDCT Records	258

Example 6. Printing the CDCT File Contents.	259
Chapter 21: PWXUCREG - Capture Registration Suspend Utility.....	262
PWXUCREG Utility Overview.	262
PWXUCREG Usage Considerations.	263
Supported Operating Systems for the PWXUCREG Utility.	264
Suspending Change Capture for Registered Sources Temporarily.	264
General Syntax for PWXUCREG Commands.	265
Summary of PWXUCREG Commands.	266
Global SET_CONTROL_VALUE Parameters.	271
Registration-specific Command Parameters.	274
Running the PWXUCREG Utility.	276
Examples of PWXUCREG Utility Commands.	276
Example 1. Suspending a Capture Registration.	276
Example 2. Reactivating a Capture Registration.	278
Chapter 22: PWXUCRGP - Capture Registrations Print Utility.....	280
PWXUCRGP Utility Overview.	280
Supported Operating Systems for the PWXUCRGP Utility.	280
Control Statement Syntax for the PWXUCRGP Utility.	281
PWXUCRGP Parameters.	281
Running the PWXUCRGP Utility.	284
PWXUCRGP Report Levels of Detail.	284
Single-Line Report Content.	284
Summary Report Content.	285
Column-Level Report Content.	287
Examples of PWXUCRGP Reports.	290
Example 1. Single-Line Report.	290
Example 2. Summary Report.	291
Example 3. Column-Level Report.	293
Chapter 23: PWXUDMX - Data Maps Update Time ECSA Memory Utility.....	296
PWXUDMX Utility Overview.	296
Supported Operating Systems for the PWXUDMX Utility.	296
Running the PWXUDMX Utility on z/OS.	297
PWXUDMX Commands and Parameters.	297
CREATE_ECSA Command.	298
DECREMENT_FILE_COUNT Command.	298
DELETE_ECSA Command.	298
DISPLAY_ECSA Command.	298
DUMP_ECSA Command.	298
INCREMENT_FILE_COUNT Command.	299

Chapter 24: PWXUGSK - SSL Reporting Utility for z/OS.....	300
PWXUGSK Utility Overview.	300
Supported Operating Systems for the PWXUGSK Utility.	300
PWXUGSK Control Statement Syntax.	301
PWXUGSK Commands and Parameters.	301
PING Command.	301
REPORT_CERTIFICATES Command.	301
REPORT_CIPHERS.	302
REPORT_ERROR_CODES Command.	302
Running the PWXUGSK Utility.	303
PWXUGSK Utility Reports.	304
Certificates Report.	304
Ciphers Report.	307
Error Codes Report.	311
 Chapter 25: PWXUMAP - Map List Utility.....	 313
PWXUMAP Utility Overview.	313
Supported Operating Systems for the PWXUMAP Utility.	314
General Syntax for the PWXUMAP Utility.	314
PWXUMAP Commands and Parameters.	314
PWXUMAP Global Parameters.	314
Summary of Commands and Parameters.	317
DTLDESCRIBE Command.	318
LISTMAPS Command.	319
LISTSCHEMAS Command.	320
PRINTMAPLINES Command.	320
PRINTMAPREPORT Command.	321
Running the PWXUMAP Utility.	322
Examples of PWXUMAP Reports.	322
Example 1. DTLDESCRIBE Report.	322
Example 2. LISTMAPS report.	326
Example 3. LISTSCHEMAS Report.	327
Example 4. PRINTMAPLINES Report.	328
Example 5. PRINTMAPREPORT Report.	330
PWXUMAP Utility Usage Notes.	330
 Chapter 26: PWXUSSL - PowerExchange SSL Reporting Utility.....	 332
PWXUSSL Utility Overview.	332
Supported Operating Systems for the PWXUSSL Utility.	333
Control Statement Syntax for PWXUSSL Commands.	333
PWXUSSL Commands and Parameters.	333
CONVERT_CERT_PKCS12_PEM Command.	333

PING Command.	334
REPORT_CERTIFICATE Command.	334
REPORT_CIPHERS Command.	335
REPORT_CODES Command.	335
REPORT_CONFIG Command.	336
REPORT_ERROR_CODES Command.	336
REPORT_VERSION Commands.	337
Running the PWXUSSL Utility.	337
PWXUSSL Utility Reports.	337
Certificate Report.	337
Ciphers Report.	338
Codes Report.	340
Configuration Report.	341
Error Codes Report.	342
Version Report.	345
Index.	347

Preface

Use the Informatica® *PowerExchange® Utilities Guide* to learn how to configure and run PowerExchange utilities. The utility programs perform a variety of functions that can help you maintain your PowerExchange environment.

Informatica Resources

Informatica provides you with a range of product resources through the Informatica Network and other online portals. Use the resources to get the most from your Informatica products and solutions and to learn from other Informatica users and subject matter experts.

Informatica Network

The Informatica Network is the gateway to many resources, including the Informatica Knowledge Base and Informatica Global Customer Support. To enter the Informatica Network, visit <https://network.informatica.com>.

As an Informatica Network member, you have the following options:

- Search the Knowledge Base for product resources.
- View product availability information.
- Create and review your support cases.
- Find your local Informatica User Group Network and collaborate with your peers.

Informatica Knowledge Base

Use the Informatica Knowledge Base to find product resources such as how-to articles, best practices, video tutorials, and answers to frequently asked questions.

To search the Knowledge Base, visit <https://search.informatica.com>. If you have questions, comments, or ideas about the Knowledge Base, contact the Informatica Knowledge Base team at KB_Feedback@informatica.com.

Informatica Documentation

Use the Informatica Documentation Portal to explore an extensive library of documentation for current and recent product releases. To explore the Documentation Portal, visit <https://docs.informatica.com>.

If you have questions, comments, or ideas about the product documentation, contact the Informatica Documentation team at infa_documentation@informatica.com.

Informatica Product Availability Matrices

Product Availability Matrices (PAMs) indicate the versions of the operating systems, databases, and types of data sources and targets that a product release supports. You can browse the Informatica PAMs at <https://network.informatica.com/community/informatica-network/product-availability-matrices>.

Informatica Velocity

Informatica Velocity is a collection of tips and best practices developed by Informatica Professional Services and based on real-world experiences from hundreds of data management projects. Informatica Velocity represents the collective knowledge of Informatica consultants who work with organizations around the world to plan, develop, deploy, and maintain successful data management solutions.

You can find Informatica Velocity resources at <http://velocity.informatica.com>. If you have questions, comments, or ideas about Informatica Velocity, contact Informatica Professional Services at ips@informatica.com.

Informatica Marketplace

The Informatica Marketplace is a forum where you can find solutions that extend and enhance your Informatica implementations. Leverage any of the hundreds of solutions from Informatica developers and partners on the Marketplace to improve your productivity and speed up time to implementation on your projects. You can find the Informatica Marketplace at <https://marketplace.informatica.com>.

Informatica Global Customer Support

You can contact a Global Support Center by telephone or through the Informatica Network.

To find your local Informatica Global Customer Support telephone number, visit the Informatica website at the following link:

<https://www.informatica.com/services-and-training/customer-success-services/contact-us.html>.

To find online support resources on the Informatica Network, visit <https://network.informatica.com> and select the eSupport option.

CHAPTER 1

Introduction to PowerExchange Utilities

This chapter includes the following topics:

- [PowerExchange Utilities Overview, 16](#)
- [PowerExchange Utilities by Operating System, 18](#)
- [PowerExchange Utilities Syntax Conventions, 19](#)
- [Environment Variable Incompatibilities Between PowerExchange and PowerCenter, 19](#)
- [PowerExchange Sample JCL, 20](#)

PowerExchange Utilities Overview

This guide is intended for PowerExchange administrators who run one or more utilities to manage their PowerExchange installations.

Each chapter provides some or all of the following reference information for a specific utility:

- The tasks that you can complete with the utility
- Any prerequisites to running the utility
- The systems on which the utility can run
- The syntax of the utility commands and parameters
- Detailed descriptions of the required and optional parameters
- How to issue the utility control statements on the various systems
- Examples of utility syntax by operating system, task, or both

PowerExchange includes the following utilities to help you manage your PowerExchange installation:

- **createdatamaps** - Data map creation utility. Use createdatamaps to generate data maps for IMS, sequential, and VSAM data sources from the Windows command line. createdatamaps is an option of the `infacmd pwx` command.
- **DTLCCADW** - Adabas PCAT file utility. Use DTLCCADW to manipulate the contents of the PCAT file.
- **DTLCUIML** - IMS log marker utility. Use DTLCUIML to define a marker for the IMS log-based ECCR in the IMS system log data set (SLDS).
- **DTLINFO** - Build information utility. Use DTLINFO to display the version, release, and build level for PowerExchange.

- DTLREXE - Remote program utility. Use DTLREXE to run programs on remote platforms.
- DTLUAPPL - Restart token utility. Use DTLUAPPL to update the CDEP file with specified applications and capture registrations.
- DTLUCBRG - Batch registration utility. Use the DTLUCBRG utility to add or modify capture registrations and extraction maps.
- DTLUCDEP - CDEP utility. Use DTLUCDEP to modify or print out the contents of the CDEP file.
- DTLUCSR2 - IDMS SR2 and SR3 records utility. Use DTLUCSR2 to determine the position of SR3 records.
- DTLUCUDB - DB2 CDC utility. Use DTLUCUDB to create a catalog snapshot to initialize the capture catalog table and to generate diagnostic information.
- DTLULCAT and DTLULOGC - IDMS log catalog utilities. Use DTLULCAT and DTLULOGC to populate the log catalog with information about the logs to process.
- DTLURDMO - Data map utility. Use DTLURDMO to migrate data maps, capture registrations and capture extraction map definitions, from one environment or location to another.
- DTLUTSK - Task control utility. Use DTLUTSK to list active tasks and stop them if required.
- EDMLUCTR - Scan and print utility for PowerExchange logs. Use EDMLUCTR to display information about the changes that are captured in the logs of the PowerExchange Logger, or to diagnose problems related to capturing changes.
- EDMXLUTL - Event marker utility. Use EDMXLUTL to create an event marker in your PowerExchange Logger.
- HOSTENT - TCP/IP Address Reporter Utility. Use HOSTENT to display the TCP/IP host name and address for a system and to diagnose problems with PowerExchange communication and licensing.
- PWXUCCLPRT - Print log summary utility. Use PWXUCCLPRT to read PowerExchange Logger for Linux, UNIX, and Windows logs and print a summary of the log content to the PowerExchange log.
- PWXUCDCT - PowerExchange Logger for Linux, UNIX, and Windows utility. Use PWXUCDCT to manage and regenerate the CDCT file, delete log files that are not referenced by CDCT records, and print reports on the CDCT file, checkpoint files, and log files.
- PWXUCREG - PowerExchange Capture Registration Suspend Utility. Use PWXUCREG to temporarily suspend change capture processing for registered sources. Later, you can use the utility to reactivate the suspended registrations to resume change capture.
- PWXUDMX - PowerExchange Data Maps Update Time ECSA memory utility. Use PWXUDMX to allocate, display, and delete ECSA memory that holds the time stamps of the latest updates to data map files and to modify the use counts of a file.
- PWXUSSL - PowerExchange SSL Reporting Utility. Use PWXUSSL to generate reports about SSL libraries and certificates on Linux, UNIX, and Windows.

PowerExchange Utilities by Operating System

The following table lists the operating systems on which each utility can run:

Utility	Linux	UNIX	Windows	z/OS	i5/OS
createdatamaps	No	No	Yes	No	No
DTLCCADW	No	No	No	Yes, for Adabas only	No
DTLCUIML	No	No	No	Yes, for IMS only	No
DTLINFO	Yes	Yes	Yes	Yes	Yes
DTLREXE	Yes	Yes	Yes	Yes	Yes
DTLUAPPL	Yes	Yes	Yes	Yes	Yes
DTLUCBRG	Yes	Yes	Yes	Yes	Yes
DTLUCDEP	Yes	Yes	Yes	Yes	Yes
DTLUCSR2	No	No	No	Yes, for IDMS CDC only	No
DTLUCUDB	Yes, for DB2 only	Yes, for DB2 only	Yes, for DB2 only	No	No
DTLULCAT and DTLULOGC	No	No	No	Yes, for IDMS Log-Based CDC only	No
DTLURDMO	Yes	Yes	Yes	Yes	Yes
DTLUTSK	Yes	Yes	Yes	Yes	Yes
EDMLUCTR	No	No	No	Yes	No
EDMXLUTL	No	No	No	Yes	No
HOSTENT	Yes	Yes	No	Yes	Yes
PWXUCCLPRT	Yes	Yes	Yes	No	No
PWXUCDCT	Yes	Yes	Yes	No	No
PWXUCREG	No	No	No	Yes	No
PWXUDMX	No	No	No	Yes	No
PWXUSSL	Yes	Yes	Yes	No	No

PowerExchange Utilities Syntax Conventions

This guide uses the following syntax conventions for the utility commands and parameters:

- All UPPERCASE letters are used for most command names and for most parameter names, regardless of the type of platform. However, positional parameters for which you enter a specific value are shown in lowercase and italics, for example, *instance*.
- Square brackets [] indicate optional parameters. You can consider any parameters without these brackets to be required.
- A vertical bar | separates alternative options of which one can be entered for a parameter.
- Single braces { } enclose alternative entries. Use only one of the entries. Do not type the braces when you enter the option.
- Underlining indicates the default option for a parameter, if available.
- Italics indicate a variable or positional parameter for which the value varies.

Environment Variable Incompatibilities Between PowerExchange and PowerCenter

When PowerCenter® and PowerExchange are installed on the same Linux, UNIX, or Windows machine, in certain cases, they have conflicting requirements for the PATH and LD_LIBRARY_PATH environment variables. To run correctly in these cases, PowerExchange and PowerCenter must run in separate environments.

This requirement applies when the PowerCenter Integration Service or PowerCenter Repository Service runs on the same machine as one of the following PowerExchange components:

- PowerExchange Listener
- PowerExchange Logger for Linux, UNIX, and Windows
- PowerExchange Navigator
- Any PowerExchange utility except the createdatamaps utility

The following table describes the restrictions that apply to the PATH and LD_LIBRARY_PATH variables in the PowerExchange and PowerCenter environments:

Environment	PATH	LD_LIBRARY_PATH
PowerExchange	\$INFA_HOME must not precede \$PWX_HOME. Otherwise, you cannot start the PowerExchange Listener or Logger from the command line.	LD_LIBRARY_PATH must not contain an entry for PowerCenter. This requirement ensures that PowerExchange utilities pick up their libraries from \$PWX_HOME only.
PowerCenter	The \$PWX_HOME entry must not precede the \$INFA_HOME entry.	The \$LD_LIBRARY_PATH variable definition must include both \$INFA_HOME and \$PWX_HOME, and \$INFA_HOME must precede \$PWX_HOME. For example: <code>\$INFA_HOME/server/bin:\$PWX_HOME: \$LD_LIBRARY_PATH</code>

To set the correct environment for PowerExchange or PowerCenter instances on the same machine, use one of the following strategies:

- Always start PowerExchange and PowerCenter using separate user accounts, and set the environment variables appropriately for each account.
- Run the `pwxssettask.sh` or `pwxssettask.bat` script each time you start a PowerExchange component.

Setting the PowerExchange Environment and Starting a Utility

If you run PowerExchange and PowerCenter using the same user account and need to create separate environments for PowerExchange and PowerCenter, run the `pwxssettask.sh` or `pwxssettask.bat` script to start a PowerExchange utility. The script sets the `PATH` and `LD_LIBRARY_PATH` variables correctly for PowerExchange.

To set the PowerExchange environment and start a PowerExchange utility on Linux or UNIX, issue the following command:

```
pwxssettask.sh utility_name parameter_list
```

To set the PowerExchange environment and start a PowerExchange utility on Windows, issue the following command:

```
pwxssettask utility_name parameter_list
```

parameter_list represents a list of expressions in the form *parameter=value*. When you run the script on Windows, you must include each *parameter=value* expression in double quotation marks. On Linux and UNIX, the quotation marks are optional.

For example, to start the DTLUCBRG utility on Windows:

```
pwxssettask dtlucbrg "cs=filename"
```

PowerExchange Sample JCL

When you install PowerExchange on z/OS, you install sample JCL to the `HLQ.RUNLIB` library.

If you chose to select the **Delete Install Members** option on the **Select Additional Parameters** tab of the z/OS Installation Assistant, the installation process moves the sample JCL to the `HLQ.DTLEXP` library.

CHAPTER 2

createdatamaps - Data Map Creation Utility

This chapter includes the following topics:

- [createdatamaps Utility Overview, 21](#)
- [Requirements and Considerations for COBOL Copybooks and DBDs, 22](#)
- [createdatamaps Command Syntax, 22](#)
- [Creating, Editing, and Testing a Data Map, 24](#)
- [Control Files for Data Map Generation, 25](#)
- [Schema File for Describing Control Files, 29](#)
- [Schema File Reference, 31](#)
- [Log File for Data Maps Creation Utility, 46](#)
- [Redefinitions and Record IDs in COBOL Copybooks, 48](#)
- [Copybook and DBD Metadata for IMS Data Maps, 54](#)
- [Column Name Filters, 55](#)
- [Unavailable Data Map Properties, 56](#)
- [Examples, 57](#)

createdatamaps Utility Overview

Use the createdatamaps utility to generate data maps for IMS, sequential, and VSAM data sources from the Windows command line. To determine the structure of the data map, the utility imports metadata from COBOL copybooks and IMS DBDs. The utility provides an alternative to using the PowerExchange Navigator in certain cases and allows you to generate or regenerate data maps noninteractively.

Also, for sequential and VSAM data sources on z/OS, the utility can find record ID fields. This feature is useful if the COBOL copybook includes REDEFINE statements or multiple 01-level records and includes one or more record ID fields. The utility reads the COBOL copybook and the data files that you specify in the control file and finds likely RID fields and the data values that they contain. For each record layout that matches all of the data records that have a given RID value, the utility creates a table and a record in the data map and assigns a data value to the RID field.

To run the utility, use the infacmd pwx createdatamaps command. The Informatica services or the Informatica Client must be installed on the Windows machine on which you run the command.

You can create multiple data maps per run, but they must all be of the same data source type.

Use the `createdatamaps` utility to create new data maps only. Do not use the utility to modify data maps that are already in use.

Note: If the command fails with a Java memory error, increase the system memory available for `infacmd`. To increase the system memory, set the `-Xmx` value in the `ICMD_JAVA_OPTS` environment variable. For more information, see the *Informatica Command Reference*.

Requirements and Considerations for COBOL Copybooks and DBDs

The `createdatamaps` utility imports metadata from COBOL copybooks and DBDs.

The following requirements and considerations apply:

- You can import metadata from only one COBOL copybook for each sequential or VSAM data map that you create. You can import metadata from one COBOL copybook for each segment in an IMS data map.
- If the copybook for a VSAM or sequential data map includes multiple 01-level records, the utility creates one record and one table in the data map for each 01-level record in the copybook. If the copybook for an IMS data map includes multiple 01-level records, the utility creates a record and a table for the first level-01 record in the copybook only.
- You can import metadata from only one DBD for each IMS data map that you create. You can optionally overlay the DBD metadata with metadata from one COBOL copybook for each segment that the DBD defines.
- For IMS data maps, if the DBD or COBOL copybook overlay includes field redefinitions, the utility creates a record only for the first combination of redefined fields.

For additional requirements and limitations that apply to finding record IDs in COBOL copybooks, see [“Requirements and Limitations for Finding RID Fields” on page 50](#).

createdatamaps Command Syntax

The `infacmd pwx createdatamaps` command uses the following syntax:

```
infacmd pwx createdatamaps
[<-pwxLocation|-loc> pwx_location]
[<-pwxUserName|-pun> pwx_user_name]
[<-pwxPassword|-ppd> pwx_password]
[<-pwxEncryptedPassword|-epwd> pwx_encrypted_password]
[<-datamapOutputDir|-dod> datamap_output_directory]
[<-replace|-r> replace_existing_datamaps]
<-controlFile|-cf> file_path_for_control_file
[<-logFile|-lf> file_path_for_log_file]
```

[<-verbosity|-v> logging_verbosity]

The following table describes infacmd pwx createdatamaps options and arguments:

Option	Argument	Description
-pwxLocation -loc	pwx_location	Optional. The location of the data source as specified in a NODE statement in the PowerExchange dbmover configuration file. If pwxLocation is not specified, the createdatamaps utility accesses the copybook and DBD metadata on the local file system. If you configure the control file to find record IDs, pwxLocation is required.
-pwxUserName -pun	pwx_user_name	Optional. The user ID for connecting to the PowerExchange Listener, if pwxLocation is specified.
-pwxPassword -ppd	pwx_password	Optional. Password for connecting to the PowerExchange Listener, if pwxLocation is specified. Instead of a password, you can enter a valid PowerExchange passphrase. Passphrases for accessing a PowerExchange Listener on z/OS can be from 9 to 128 characters in length and can contain the following characters: <ul style="list-style-type: none"> - Uppercase and lowercase letters - The numbers 0 to 9 - Spaces - The following special characters: ' - ; # \ , . / ! % & * () _ + { } : @ < > ? <p>Note: The first character is an apostrophe.</p> <p>Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.</p> <p>If a passphrase contains spaces, you must enclose it with double-quotation marks ("), for example, "This is an example passphrase". If a passphrase contains special characters, you must enclose it with triple double-quotation characters ("""), for example, """"This passphrase contains special characters ! % & *."""". If a passphrase contains only alphanumeric characters without spaces, you can enter it without delimiters.</p> <p>Note: On z/OS, a valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.</p> <p>To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the <i>PowerExchange Reference Manual</i>.</p>

Option	Argument	Description
-pwxEncryptedPassword -epwd	pwx_encrypted_password	Optional. Encrypted password for connecting to the PowerExchange Listener, if pwxLocation is specified. If the PowerExchange Listener runs on a z/OS or i5/OS system, you can enter an encrypted PowerExchange passphrase instead of an encrypted password. Do not encrypt a passphrase that contains characters that are not valid, such as double-quotation marks, single quotation marks, or currency symbols.
-datamapOutputDir -dod	datamap_output_directory	Optional. The local file directory to which to write the output data maps. Default is the current working directory.
-replace -r	replace_existing_datamaps	Optional. Specifies whether to replace existing data maps. If replace=Y, replaces any data maps in datamap_output_directory that have the same name as the data map that you are creating. If replace=N, skips the creation of a data map if a data map with the same name already exists in datamap_output_directory. Default is N.
-controlFile -cf	file_path_for_control_file	Required. Path and file name of the control file that controls data map generation.
-logFile -lf	file_path_for_log_file	Optional. Path and file name of the output log file. Default is STDOUT.
-verbosity -v	logging_verbosity	Optional. Verbosity for log files. Default is INFO. Valid values: - DEBUG. Most detailed logging. Might show stack traces. - INFO. Informational messages. - WARN. Indicates a potential problem. - ERROR. Indicates a failure. Processing continues. - FATAL. Indicates a fatal condition. Process terminates.

The PowerExchange node name and credentials are optional. If you do not include the pwxLocation option, the command accesses the local file system directly to read metadata. In this case, PowerExchange does not need to be installed on the machine on which you run createdatamaps.

Creating, Editing, and Testing a Data Map

To create, edit, and test a data map, perform the following high-level tasks:

1. If PowerExchange is not installed on the local machine, where you will run the createdatamaps command, copy the copybooks and DBDs to the local machine.

If you configure the control file to find record ID fields, the copybooks must reside on the source z/OS system, and PowerExchange must be installed on the machine on which you run the createdatamaps command.

2. Create the control file.

You can create a new file using an XML editor. Or you can open a sample control file in an XML editor, rename the file, and edit the file as needed.

3. Run the `infacmd pwx createdatamaps` command from the command line on any machine that can access the Informatica domain.
4. Review the output log. If errors occur, correct them and run the command again.
5. Copy the data map to the PowerExchange Navigator machine.
6. Open the data map in the PowerExchange Navigator.

Tip: If the PowerExchange Navigator is already open when you copy the data map, select **File > Refresh** to refresh the list of data maps.

7. If required, edit the data map.

For example, you might need to delete unused records, assign record ID values, or assign other properties that you cannot assign with the `createdatamaps` utility.

If you configured the control file to find record ID fields, review the record types that the utility created and the record ID values that the utility assigned. Edit these results if necessary.

8. Perform a database row test.
9. Make additional edits and perform additional database row tests, as needed.
10. Send the data map to the remote node.

Control Files for Data Map Generation

A control file is an XML file that defines the options and values that the `createdatamaps` utility uses to create one or more data maps. A control file provides a simple way to define parameters and helps to make the bulk import creation process easily repeatable.

When you run the `createdatamaps` command, provide the name of the control file as a parameter.

Each control file must conform to the schema that is installed with the Informatica Client and with Informatica services. For more information, see [“Schema File for Describing Control Files” on page 29](#) and [“Schema File Reference” on page 31](#).

Each control file can specify only one data source type.

After you create the control file and run the `createdatamaps` command, you might need to edit the data map in the PowerExchange Navigator.

Example control files are provided with the product. For more information, see [“Examples” on page 57](#).

Rules for Control Files

Observe the following rules when you create a control file:

- All XML elements are case sensitive. For each element, follow the capitalization that is defined in the schema.
- Elements must appear in the control file in the order in which they are defined in the schema. Your XML editor should indicate the allowable elements at each point in the control file.

- Observe the properties that are defined for each element in the schema file. For each element, the schema file might define the following properties, depending on the data type:
 - Name
 - Cardinality, that is, the minimum and maximum number of occurrences
 - Type
 - Valid values
 - Minimum and maximum length

Control File Structure

The XML control files for sequential, VSAM, and IMS data maps have hierarchies that allow certain elements at specific points.

Each control file includes the following elements:

- DatamapGeneration root element. Required.
- imsGen, seqGen, or vsamGen element that is a child of the DatamapGeneration element. Required.
- Global elements that apply to all data maps that the control file defines. These global elements are children of the imsGen, seqGen, or vsamGen element. Optional.
- DatamapInstances element. This element is a child of the imsGen, seqGen, or vsamGen element. Required.
- One imsDatamapInstance, seqDatamapInstance, or vsamDatamapInstance element for each data map that the control file defines. These elements are children of the DatamapInstances element. At least one element instance is required.

For each data map, the imsDatamapInstance, seqDatamapInstance, or vsamDatamapInstance element inherits and optionally overrides the global settings.

The following control file hierarchies do not show the required syntax, whether the element is required or optional, the number of times the element can occur, or whether a choice of only one of multiple elements can appear. For this information, see [“Schema File Reference” on page 31](#) or see the DatamapGeneration.xsd schema file.

Control File Hierarchy for Sequential Data Maps

Use the following control file hierarchy for sequential data maps:

```
DatamapGeneration
  seqGen
    globalCopybookParserConfig
      startColumn
      endColumn
      maxRedefines
      decimalPointIsComma
    cacheConfig
      cachePath
      flushDataMode
    globalGenConfig
      schemaName
      datamapName
      datamapRecordName
      filterColumnGroup
      filter
      columnName
      exclude
      tableName
    findRecordIds
```

```

    excludeUnmatchedRecords
globalSeqProperties
skipRecordCount
ridConfig
    readRecordLimit
    recordTypeLimit
    fieldWidth
    matchFieldWidth
    fieldOffset
seqFileName
zosPath
windowsPath
as400Pth
unixPath
datamapInstances
seqDatamapInstance
    genConfig
        schemaName
        datamapName
        datamapRecordName
        filterColumnGroup
        filter
            columnName
            exclude
            tableName
        findRecordIds
        excludeUnmatchedRecords
importCopybookDetails
    filePath
        zosPath
        windowsPath
        as400Path
        unixPath
    parserConfig
        startColumn
        endColumn
        maxRedefines
        decimalPointIsComma
datamapProperties
    seqFileName
    zosPath
    windowsPath
    as400Pth
    unixPath
    skipRecordCount
    ridConfig
        readRecordLimit
        recordTypeLimit
        fieldWidth
        matchFieldWidth
        fieldOffset

```

Control File Hierarchy for VSAM Data Maps

Use the following control file hierarchy for VSAM data maps:

```

DatamapGeneration
    vsamGen
        globalCopybookParserConfig
            startColumn
            endColumn
            maxRedefines
            decimalPointIsComma
        cacheConfig
            cachePath
            flushDataMode
        globalGenConfig
            schemaName
            datamapName
            datamapRecordName

```

```

    filterColumnGroup
      filter
        columnName
        exclude
        tableName
    findRecordIds
    excludeUnmatchedRecords
  globalVsamProperties
    skipRecordCount
    ridConfig
      readRecordLimit
      recordTypeLimit
      fieldWidth
      matchFieldWidth
      fieldOffset
    vsamFileName
      zosPath
      windowsPath
      as400Path
      unixPath
  globalMapType
  datamapInstances
    vsamDatamapInstance
      genConfig
        schemaName
        datamapName
        datamapRecordName
        filterColumnGroup
          filter
            columnName
            exclude
            tableName
          findRecordIds
          excludeUnmatchedRecords
        importCopybookDetails
          filePath
            zosPath
            windowsPath
            as400Path
            unixPath
          parserConfig
            startColumn
            endColumn
            maxRedefines
            decimalPointIsComma
        datamapProperties
          vsamFileName
          zosPath
          windowsPath
          as400Path
          unixPath
          skipRecordCount
          ridConfig
            readRecordLimit
            recordTypeLimit
            fieldWidth
            matchFieldWidth
            fieldOffset

```

Control File Hierarchy for IMS Data Maps

Use the following control file hierarchy for IMS data maps:

```

  DatamapGeneration
    imsGen
      globalCopybookParserConfig
        startColumn
        endColumn
        maxRedefines
        decimalPointIsComma

```

```

globalGenConfig
  schemaName
  datamapName
  datamapRecordName
  filterColumnGroup
    filter
      columnName
      exclude
      tableName
  createTablesForHierPath
globalImsProperties
  mapType
  imsSSID
  pcbNumber
  psbName
  pcbName
datamapInstances
  imsDatamapInstance
    genConfig
      schemaName
      datamapName
      datamapRecordName
      filterColumnGroup
        filter
          columnName
          exclude
          tableName
      createTablesForHierPath
importDBDDetails
  filePath
  zosPath
  windowsPath
  as400Path
  unixPath
overlayDetails
  nativeRecordName
  overlayCopybookDetails
    filePath
    zosPath
    windowsPath
    as400Path
    unixPath
  parserConfig
    startColumn
    endColumn
    maxRedefines
    decimalPointIsComma
datamapProperties
  mapType
  imsSSID
  pcbNumber
  psbName
  pcbName

```

Schema File for Describing Control Files

Each control file must conform to the schema that is installed with the Informatica Client and with Informatica services.

The schema file, named `DatamapGeneration.xsd`, is provided in the following directories on the machines where the Informatica Client and the Informatica services are installed:

- `Informatica_client_installation_directory\clients\DeveloperClient\osgi_mf_plugins\jars\resources\`
- `Informatica_services_installation_directory\pwxmfplugins\resources\`

The schema consists of definitions of complex types and elements, where a complex type consists of multiple elements.

Complex Type Definitions in the Schema File

For each complex type, the schema file includes the following information:

- Name
- Base type, for complex types that inherit properties from a base type
- Definitions of each child element

Element Definitions in the Schema File

For each element, the schema file might include the following information, depending on the data type:

- Name
- Cardinality, that is, the minimum and maximum number of occurrences
- Type
- Documentation
- Valid values
- Minimum and maximum length

Using the Schema File to Determine the Control File Hierarchy

You can open the schema file in an XML editor to determine the allowable hierarchy of elements in a control file. You can also derive this information from the schema file.

For example, the root element in the schema is `DatamapGeneration`. The definition of the `DatamapGeneration` complex type specifies a choice of one of the following child elements:

- `seqGen`, of type `SEQGen`
- `vsamGen`, of type `VSAMGen`
- `imsGen`, of type `IMSGen`

The definition of the `SEQGen` type specifies that it is an extension of the `GenBase` base type, which includes the following elements: `globalCopybookParserConfig` and `cacheConfig`. Elements of type `SEQGen` include the elements defined by the `GenBase` type in addition to the following elements:

- `globalGenConfig`, of type `SEQGenConfig`
- `globalSeqProperties`, of type `SEQDatamapProperties`
- `datamapInstances`, which consists of one element, `seqDatamapInstance`, that can occur one or more times

The control file for generating SEQ data maps therefore has the following high-level structure:

```
DatamapGeneration
  seqGen
    globalCopybookParserConfig (optional)
    cacheConfig (optional)
    globalGenConfig (optional)
    globalSeqProperties (optional)
    datamapInstances (required)
      seqDatamapInstance (at least one required)
```

You can continue in this manner to determine the complete hierarchy for SEQ control files, as well as for VSAM and IMS control files.

Schema File Reference

The following topics describe each complex type and element in the schema file, including its type, allowed values, cardinality, and description.

DatamapGeneration Complex Type

The DatamapGeneration complex type defines the top-level element in the control file.

The DatamapGeneration complex type includes the following elements:

Choice (cardinality = 1):

- SeqGen
- VSAMGen
- IMSGen

Exactly one occurrence of SeqGen, VSAMGen, or IMSGen is required.

seqGen

Element for defining sequential data maps.

Type = SEQGen

vsamGen

Element for defining VSAM data maps.

Type = VSAMGen

imsGen

Element for defining IMS data maps.

Type = IMSGen

The DatamapGeneration complex type includes the required xmlSchemaVersion attribute.

xmlSchemaVersion

Latest schema version with which the XML instance is compatible.

Value = 1.0

GenBase Complex Type

The GenBase complex type is the base type for the complex types that define the sequential, VSAM, and IMS data maps.

The GenBase complex type is the base type for the following complex types:

- SEQGen
- VSAMGen
- IMSGen

The GenBase complex type includes the following elements:

- globalCopybookParserConfig
- cacheConfig

globalCopybookParserConfig

Default copybook parser configuration properties that are applied at a global level.

Type = CopybookParserConfig

Cardinality = 0 to 1

cacheConfig

Controls the data cache on disk.

Type = CacheConfig

Cardinality = 0 to 1

DataConfigBase Complex Type

The DataConfigBase complex type defines configuration properties for reading data records. It is the base type for the RIDConfig complex type.

The DataConfigBase complex type includes the following elements:

readRecordLimit

Maximum number of data records to read from each data file. A value of 0 means that no limit exists. The utility reads all records.

Type = integer

Range = 0 to 2147483647

Default = 10000

Cardinality = 0 to 1

SEQGen Complex Type

The SEQGen complex type defines one or more data maps for sequential data sources.

The SEQGen complex type is an extension of the GenBase complex type. SEQGen extends GenBase with the following elements:

- globalGenConfig
- globalSeqProperties
- datamapInstances

globalGenConfig

Default generator configuration properties that are applied at a global level.

Type = SEQGenConfig

Cardinality = 0 to 1

globalSeqProperties

Global defaults for SEQ data map properties.

Type = SEQDatamapProperties

Cardinality = 0 to 1

datamapInstances

Element that contains one or more occurrence of SEQ data map instance details.

Type = Container of SeqDatamapInstance elements

Cardinality = 1

VSAMGen Complex Type

The VSAMGen complex type defines one or more data maps for VSAM data sources.

The VSAMGen complex type extends the GenBase complex type with the following elements:

- globalGenConfig
- globalVSAMProperties
- globalMapType
- datamapInstances

globalGenConfig

Default generator configuration properties that are applied at a global level.

Type = VSAMGenConfig

Cardinality = 0 to 1

globalVsamProperties

Global defaults for VSAM data map properties.

Type = VSAMDatamapProperties

Cardinality = 0 to 1

globalMapType

Global value for data map access method type.

Valid values: ESDS, KSDS, RRDS

Default: KSDS

Cardinality = 0 to 1

datamapInstances

Element that contains one or more occurrence of VSAM data map instance details.

Type = Container of vsamDatamapInstance elements

Cardinality = 1

IMSGen Complex Type

The IMSGen complex type defines one or more data maps for IMS data sources.

The IMSGen complex type extends the GenBase complex type with the following elements:

- globalGenConfig
- globalIMSPProperties
- datamapInstances

globalGenConfig

Default generator configuration properties that are applied at a global level.

Type = IMSGenConfig

Cardinality = 0 to 1

globalIMSProperties

Global defaults for IMS data map properties.

Type = IMSDatamapProperties

Cardinality = 0 to 1

datamapInstances

Element that contains one or more occurrences of IMS data map instance details.

Type = Container of IMSDatamapInstance elements

Cardinality = 1

GenConfigBase Complex Type

The GenConfigBase complex type is the base type for configuration properties for the data map generation process. For example, these properties affect how the generated data maps or their records or tables are named.

The GenConfigBase complex type is the base type for the following complex types:

- SEQGenConfig
- VSAMGenConfig
- IMSGenConfig

The GenConfigBase complex type includes the following elements:

- schemaName
- datamapName
- datamapRecordName
- filterColumnGroup

schemaName

Data map schema name.

Type = string, length = 1 to 10

Default = SCHEMA

Cardinality = 0 to 1

datamapName

Data map name. Used as a data map name prefix when used at a global level. When multiple data maps are created, an integer is appended to the prefix to form the data map name.

Type = string, length = 1 to 10

Default = MAP

Cardinality = 0 to 1

datamapRecordName

Prefix for names of records in the data map. When multiple records are created, an integer is appended to the prefix to form the record name.

Type = string, length = 1 to 256

Default = name of native import object name, such as a COBOL 01-level record name or a DBD segment name.

Cardinality = 0 to 1

filterColumnGroup

Filters columns in data maps based on column name.

Type = FilterColumnGroup

Cardinality = 0 to 1

SEQGenConfig Complex Type

The SEQConfigBase complex type defines configuration properties for the data map generation process for sequential data sources.

The SEQGenConfig complex type is an extension of the GenConfigBase complex type. It includes the following additional element:

- findRecordIds
- excludeUnmatchedRecords

findRecordIds

Enables or disables discovery of record IDs.

Type = boolean

Default = false

Cardinality = 0 to 1

excludeUnmatchedRecords

If true, generates a data map record only for those layouts for which the utility finds a valid data record ID.

Type = boolean

Default = false

Cardinality = 0 to 1

VSAMGenConfig Complex Type

The VSAMGenConfig complex type defines configuration properties for the data map generation process for VSAM data sources.

The VSAMGenConfig complex type is an extension of the GenConfigBase complex type. It includes the following additional element:

- findRecordIds
- excludeUnmatchedRecords

findRecordIds

Enables or disables discovery of record IDs.

Type = boolean

Default = false

Cardinality = 0 to 1

excludeUnmatchedRecords

If true, generates a data map record only for those layouts for which the utility finds a valid data record ID.

Type = boolean

Default = false

Cardinality = 0 to 1

IMSGenConfig Complex Type

The IMSConfigBase complex type defines configuration properties for the data map generation process for IMS data sources.

The IMSGenConfig complex type is an extension of the GenConfigBase complex type. It includes the following additional elements:

- CreateTablesForHierPath

CreateTablesForHierPath

Whether to create complex tables that include all of the records in the hierarchical path.

Type = boolean

Default = true

Cardinality = 0 to 1

DatamapPropertiesBase Complex Type

The DatamapPropertiesBase complex type defines common data map properties.

The DatamapPropertiesBase complex type is the base type for the following complex types:

- SEQDatamapProperties
- VSAMDatamapProperties
- IMSDatamapProperties

The DatamapPropertiesBase complex type does not include any elements.

ParserConfigBase Complex Type

The ParserConfigBase complex type includes common parser configuration properties.

The ParserConfigBase complex type is the base type for the following complex types:

- CopybookParserConfig

ParserConfigBase does not include any elements.

CopybookParserConfig Complex Type

The CopybookParserConfig complex type includes parser configuration properties for copybooks.

The CopybookParserConfig complex type is an extension of the ParserConfigBase complex type and includes the following elements:

- startColumn
- endColumn
- maxRedefines
- decimalPointIsComma

startColumn

Starting column of data to be parsed.

Type = integer, range = 1 to 999

Default = 7

Cardinality = 0 to 1

endColumn

Ending column of data to be parsed.

Type = integer, range = 1 to 999

Default = 72

Cardinality = 0 to 1

maxRedefines

Maximum number of record layouts to generate from REDEFINE statements in the copybook. If the copybook overlays a DBD, maxRedefines defaults to 1 and cannot be overridden. If the copybook for a sequential or VSAM data map contains multiple 01-level records, maxRedefines applies to each 01-level record in the copybook.

Type = integer, range = 1 to 4096

Default = 10000 if the control file is configured to find RID fields, otherwise default = 1

Cardinality = 0 to 1

decimalPointIsComma

Defines whether a comma represents a decimal point character in fields that contain noninteger numbers. Set this value to match the value of the DECPOINT statement in the DBMOVER configuration file.

Type = boolean

Default = false

Cardinality = 0 to 1

CacheConfig Complex Type

The CacheConfig complex type controls the data cache on disk. You can configure CacheConfig properties at a global level but not a data map instance level.

The CacheConfig complex type includes the following elements:

cachePath

The full path to the folder for temporary working files. The cache path is written to the message log.

Type = string

Cardinality = 0 to 1

Default = *current_working_directory/temp*

flushDataMode

Specifies when to flush the cache of the data records that were downloaded from the z/OS system.

Type = string

Valid values:

- e - Flush the cache when the createdatamaps utility finishes.
- d - Flush the cache after each data map is created.

The default value of "e" allows data to be shared by multiple data map generations during one createdatamaps session.

RIDConfig Complex Type

The RIDConfig complex type defines criteria for finding record ID (RID) fields.

The RIDConfig complex type is an extension of the DataConfigBase complex type. RIDConfig extends DataConfigBase with the following elements:

- recordTypeLimit
- fieldWidth
- matchFieldWidth
- fieldOffset

The RIDConfig complex type includes the following elements:

recordTypeLimit

Maximum number of record types in the data file.

Type = integer, range = 1 to 2147483647

Default = 10

Cardinality = 0 to 1

fieldWidth

Maximum width or exact width in bytes of an RID field, depending on matchFieldWidth.

Type = integer, range = 1 to 2147483647

Default = 4

Cardinality = 0 to 1

matchFieldWidth

If true, the RID field must exactly match the fieldWidth value. If false, fieldWidth represents the maximum field width.

Type = boolean

Default = false

Cardinality = 0 to 1

fieldOffset

Byte offset of the RID field from the start of the record, beginning at offset 0. If not specified, the utility finds the RID field.

Type = integer, range = -1 to 2147483647

Default = -1, meaning not specified

Cardinality = 0 to 1

FilePath Complex Type

The FilePath complex type defines the path and name of a file on the file system. The file path can be absolute or relative to the current directory.

The FilePath complex type includes the following elements:

- Choice (cardinality = 1):
 - zosPath
 - windowsPath
 - as400Path
 - unixPath

zosPath

File path on a z/OS system.

Type = string, length = 1 to 256

Cardinality = 0 to 1

windowsPath

File path on a Windows system.

Type = string, length = 1 to 1024

Cardinality = 0 to 1

as400Path

File path on an i5/OS system.

Type = string, length = 1 to 256

Cardinality = 0 to 1

unixPath

File path on a UNIX system.

Type = string, length = 1 to 1024

Cardinality = 0 to 1

ImportMetadataBase Complex Type

The ImportMetadataBase complex type defines common metadata import properties.

ImportMetadataBase is the base type for the following complex types:

- CopybookImportMetadata
- DBDImportMetadata

ImportMetadataBase includes the following elements:

- filePath

filePath

File system location of the metadata source.

Type = FilePath

Cardinality = 1

CopybookImportMetadata Complex Type

The CopybookImportMetadata complex type defines common metadata import properties for copybooks.

CopybookImportMetadata extends the ImportMetadataBase complex type with the following elements:

- parserConfig

parserConfig

Fields related to copybook parser configuration.

Type = CopybookParserConfig

Cardinality = 0 to 1

DBDImportMetadata Complex Type

The DBDImportMetadata complex type defines common metadata import properties for DBDs.

The DBDImportMetadata complex type is an extension of the ImportMetadataBase complex type. It does not define any additional elements.

OverlayMetadata Complex Type

The OverlayMetadata complex type defines properties for overlaying metadata, such as overlaying DBD metadata with COBOL copybook metadata.

The OverlayMetadata complex type includes the following elements:

- nativeRecordName
- overlayCopybookDetails

nativeRecordName

Native name of the data map record to overlay, such as the name of the DBD segment.

Type = string, length = 1 to 256

Cardinality = 1

overlayCopybookDetails

Details about the copybook that overlays the data map record.

Type = CopybookImportMetadata

Cardinality = 1

DatamapInstanceBase Complex Type

The DatamapInstanceBase complex type defines properties for a data map instance.

DatamapInstanceBase is the base type for the following complex types:

- SEQDatamapInstance
- VSAMDatamapInstance
- IMSDatamapInstance

The DatamapInstanceBase complex type does not define any elements.

SEQDatamapProperties Complex Type

The SEQDatamapProperties complex type defines common data map properties for sequential data sources.

The SEQDatamapProperties complex type is an extension of the DatamapPropertiesBase complex type. It extends DatamapPropertiesBase with the following elements:

- seqFileName
- skipRecordCount
- ridConfig

seqFileName

Full path and file name of the sequential data set or flat file that is a data source.

Type = FilePath

Default = current Windows path with value of "file.dat"

Cardinality = 0 to 1

skipRecordCount

Specifies the number of initial records to skip when reading the data file

Type = integer, range = 0 to 2147483647

Default = 0

Cardinality = 0 to 1

ridConfig

Defines record ID (RID) configuration parameters.

Type = RIDConfig

Cardinality = 0 to 1

SEQDatamapInstance Complex Type

The SEQDatamapInstance complex type defines properties for SEQ data maps.

SEQDatamapInstance extends the DatamapInstanceBase complex type with the following elements:

- genConfig
- importCopybookDetails
- datamapProperties

genConfig

SEQ generator configuration applied at the instance level.

Type = SEQGenConfig

Cardinality = 0 to 1

importCopybookDetails

Copybook definitions for importing data map instance metadata.

Type = CopybookImportMetadata

Cardinality = 1 to unbounded

datamapProperties

Data map properties at the instance level.

Type = SEQDatamapProperties

Cardinality = 0 to 1

VSAMDatamapProperties Complex Type

The VSAMDatamapPropertiesBase complex type defines common data map properties for VSAM data sources.

The VSAMDatamapProperties complex type is an extension of the DatamapPropertiesBase complex type. It extends DatamapPropertiesBase with the following elements:

- vsamFileName
- skipRecordCount
- ridConfig

vsamFileName

Fully qualified data set name of the of VSAM source file.

Type = FilePath

Default = Current Windows path with value of "file.dat"

Cardinality = 0 to 1

skipRecordCount

Specifies the number of initial records to skip when reading the data file

Type = integer, range = 0 to 2147483647

Default = 0

Cardinality = 0 to 1

ridConfig

Defines RID configuration parameters.

Type = RIDConfig

Cardinality = 0 to 1

VSAMDatamapInstance Complex Type

The VSAMDatamapInstance complex type defines properties for VSAM data maps.

SEQDatamapInstance extends the DatamapInstanceBase complex type with the following elements:

- genConfig
- importCopybookDetails
- datamapProperties

genConfig

VSAM generator configuration applied at the instance level.

Type = VSAMGenConfig

Cardinality = 0 to 1

importCopybookDetails

Copybook definitions for importing data map instance metadata.

Type = CopybookImportMetadata

Cardinality = 1 to unbounded

datamapProperties

Data map properties at the instance level.

Type = VSAMDatamapProperties

Cardinality = 0 to 1

IMSDatamapProperties Complex Type

The IMSDatamapPropertiesBase complex type defines common data map properties for IMS data sources.

The IMSDatamapProperties complex type extends the DatamapPropertiesBase complex type with the following elements:

- mapType
- imsSSID
- Choice (cardinality = 0 to 1, default = 1):
 - pcbNumber
 - psbName, pcbName

mapType

Data map type. Represents IMS DL/1 batch or IMS ODBA.

Type = string, valid values = ODBA, DL1

Default = DL1

Cardinality = 0 to 1

imsSSID

IMS subsystem ID.

Type = string, maximum length = 4

Cardinality = 0 to 1

pcbNumber

PCB number for the database. Optional field for DL/1 data maps at the instance level.

Type = string

Default = 1

Cardinality = 0 to 1

psbName

PSB name. Optional field for ODBA data maps at the instance level.

Type = string

Default = PSBNAME

Cardinality = 0 to 1

pcbName

For the specified PSB, the named PCB that references the specified DBD. Optional field for ODBA data maps at the instance level.

Type = string

Default = PCBNAME

Cardinality = 0 to 1

IMSDatamapInstance Complex Type

The IMSDatamapInstance complex type defines properties for IMS data maps.

IMSDatamapInstance extends the DatamapInstanceBase complex type with the following elements:

- genConfig
- importDBDDetails
- overlayDetails
- datamapProperties

genConfig

IMS generator configuration applied at an instance level.

Type = IMSGenConfig

Cardinality = 0 to 1

importDBDDetails

DBD definitions for importing data map instance metadata.

Type = DBDImportMetadata

Cardinality = 1

overlayDetails

Metadata to overlay imported records.

Type = OverlayMetadata

Cardinality = 0 to unbounded

datamapProperties

Data map properties at the instance level.

Type = IMSDatamapProperties

Cardinality = 0 to 1

FilterColumnGroup Complex Type

The FilterColumnGroup complex type defines filtering criteria for a table or set of tables.

The FilterColumnGroup complex type includes the following elements:

Filter

Defines the filter criteria.

Type = Filter

Cardinality = 0 to 1

Filter Complex Type

The Filter complex type defines the filtering criteria for columns and optionally for tables.

columnName

Column name to be filtered.

The columnName element accepts asterisks (*) and question marks (?) as wildcard characters. An asterisk represents one or more matching characters. A question mark represents a single matching character.

Type = string, length = 1 to 128

Default = FILLER*

Cardinality = 0 to 1

exclude

Whether to include or exclude column names that match the criteria.

Type = boolean

Default = true

Cardinality = 0 to 1

tableName

Name of table to which the criteria apply. If tableName is not specified, the filter criteria apply to all tables.

Type = string, length = 1 to 128

Cardinality = 0 to 1

Note: If the `globalGenConfig` element in the control file includes a `datamapRecordName` element that overrides the original table names, use the new table names in the `tableName` element within the `filter` element in the control file. For example, suppose the control file includes the following lines:

```
<globalGenConfig>
  <schemaName>SEQ055</schemaName>
  <datamapName>SEQ</datamapName>
  <datamapRecordName>RECORD</datamapRecordName>
</globalGenConfig>
```

The following lines show how to specify the new table names in the `tableName` element within the `filter` element:

```
<filterColumnGroup>
  <filter>
    <columnName>TABLE*</columnName>
    <exclude>false</exclude>
    <tableName>RECORD*</tableName>
  </filter>
</filterColumnGroup>
```

Log File for Data Maps Creation Utility

The `createdatamaps` utility writes informational, warning, and error messages to the log file that you specify when you run the command. If you do not specify a log file, the output is sent to the console.

After creating a data map, the utility writes informational messages to the log file. The messages report basic statistics such as the number of records and fields per data map.

Before executing the control file, the utility checks for syntax errors. If the utility encounters an error, it stops after the first error and reports the error in the log file.

The `createdatamaps` utility checks for the following kinds of syntax errors:

- Missing mandatory elements
- General syntax errors, such as unexpected property names
- Illegal values, where the element definition provides enumerated values
- Invalid characters in names
- Maximum length of name fields exceeded

Each message in the log file might have one of the following prefixes, depending on the module that reported it:

Prefix	Module Description
PWXCMD	Infacmd user interface
PWXLog	Log file implementation for data map generation
MDO	Component that interprets the XML control file and acts upon it
PWXNative	PowerExchange connection support for communication with the PowerExchange Listener
MDAdapter	Component that reads the metadata

Prefix	Module Description
Parser	Component that parses the COBOL, DBD, or VSAM metadata text files and turns them into Informatica object models
JDMX2	Component that interprets the Informatica object models and writes them to PowerExchange data maps

For examples of log files, see [“Examples” on page 57](#).

For error message descriptions, see the *Informatica Message Reference*.

JAXB Error Messages

In addition to the error messages that PowerExchange generates, the createdatamaps log file might include Java Architecture for XML Binding (JAXB) error messages. These messages might appear standalone or wrapped in PowerExchange message MDO_34611.

The following table shows some of the error conditions that result in a JAXB error message:

Error Condition	JAXB Error Message
An element is spelled incorrectly	The element type " <i>element_name</i> " must be terminated by the matching end-tag "</ <i>element_name</i> ".
An incorrect data type or value is specified for an element.	Not a Valid Datatype: <i>value</i> cvc-datatype-valid.1.2.1: ' <i>value</i> ' is not a valid value for 'Datatype name' cvc-type.3.1.3: The value ' <i>value</i> ' of element ' <i>element_name</i> ' is not valid.
An end tag is missing.	The element type " <i>element_name</i> " must be terminated by the matching end-tag "</ <i>element_name</i> >".
Data is specified outside of the element tags.	Element ' <i>element_name</i> ' cannot have character [children], because the type's content type is element-only.
The DatamapGeneration tag does not include the xmlSchemaVersion and xmlns attributes.	Unexpected element (uri:" <i>uri</i> ", local:" <i>element_name</i> "). Expected elements are <i>element_list</i> .
An element is specified in an incorrect location.	cvc-complex-type.2.4.a: Invalid content was found starting with element ' <i>element_name</i> '. One of '{ <i>element_list</i> }' is expected.
The order of elements is rearranged.	Invalid content was found starting with element ' <i>element1_name</i> '. No child element is expected at this point.. <i>element2_name</i>
A reserved character, such as a greater-than or less-than sign, is specified as part of a value for an element.	Value ' <i>value</i> ' is not facet-valid with respect to pattern '[a-zA-Z][a-zA-Z0-9_]*' for type '#type'.. XML parser message: SEVERITY: 2, MESSAGE: cvc-type.3.1.3: The value ' <i>value</i> ' of element ' <i>element_name</i> ' is not valid..

Redefinitions and Record IDs in COBOL Copybooks

COBOL copybooks can define multiple record layouts. The layout of each data record is often determined by a record ID (RID) field.

The following types of redefinitions in a COBOL copybook define multiple record layouts:

- REDEFINE statements
- Multiple 01 levels that define multiple record types

The `createdatamaps` utility can generate data maps for COBOL copybooks that include REDEFINE statements, multiple 01 levels, or both. And you can configure the control file so that the utility finds RID fields and associates RID values with data records.

How the `createdatamaps` Utility Creates Records for Redefined Fields and Groups

The `createdatamaps` utility builds an internal model that represents all of the combinations of redefined fields and groups.

For example, consider the following record from a COBOL copybook:

```
01 MASTER-REC.  
    05 MASTER-DATE.  
        07 some fields  
    05 MASTER-DOB REDEFINES MASTER-DATE.  
        07 some fields  
    05 OTHER-DATE.  
        07 some fields  
    05 OTHER-DOB REDEFINES OTHER-DATE.  
        07 some fields
```

The utility creates an internal model with four layouts. the layouts correspond to the following combinations of fields:

- MASTER-DATE / OTHER-DATE
- MASTER-DATE / OTHER-DOB
- MASTER-DOB / OTHER-DATE
- MASTER-DOB / OTHER-DOB

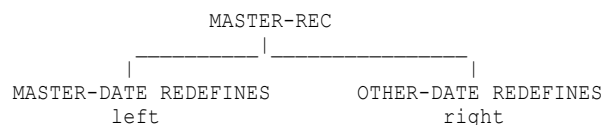
The utility creates a record and table for each of the first *maxRedefines* layouts in the model. For more information, see [“Use of the maxRedefines Element to Limit the Number of Records” on page 48](#).

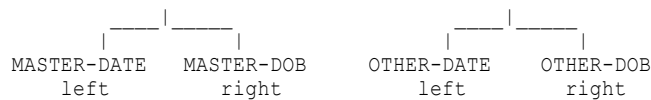
Use of the `maxRedefines` Element to Limit the Number of Records

Use the `maxRedefines` element to limit the number of layouts that the `createdatamaps` utility generates when it parses a copybook that contains REDEFINE statements. The utility writes only the first *maxRedefines* records to the data map.

If the copybook includes multiple 01 levels, `MaxRedefines` applies to each 01 level.

To determine the order of records, the utility builds a hierarchical internal model, such as the following one:





When the utility begins reading records from the COBOL copybook, it selects the left-most redefine in each branch. In this example, the utility selects the combination of MASTER-DATE / OTHER-DATE. The utility then progresses through the redefines of the right-most branch, that is, MASTER-DATE / OTHER-DOB, until all combinations are added to the model. The utility returns to the next redefine in the left branch, that is, MASTER-DOB. The utility combines MASTER-DOB with each redefine from the right branch, that is, MASTER-DOB / OTHER-DATE and then MASTER-DOB / OTHER-DOB.

Because this process can lead to a large number of records, you can include the `maxRedefines` element in the control file to limit the number of layouts that the utility processes to the first *maxRedefines* layouts.

After you create the data map, you can edit the data map in the PowerExchange Navigator to delete unwanted records and tables. Also, if you configured the control file to find RID fields, you can view the assigned RID values.

Redefinitions Without Record IDs in COBOL Copybooks

A COBOL copybook might include REDEFINE statements but no RID field. In this case, the `createdatamaps` utility defines one record and one table for each possible combination of redefined fields, up to the maximum number that the `maxRedefines` element specifies. By default, `maxRedefines` specifies 1. For IMS data maps, the default of 1 always applies.

If a copybook for a sequential or VSAM data map contains multiple 01-level records, `MaxRedefines` applies to each 01-level record in the copybook.

After you create the data map, you can edit the data map in the PowerExchange Navigator as needed to delete unwanted records and tables.

Redefinitions with RID Fields in COBOL Copybooks

If a COBOL copybook includes redefinitions and RID fields, you can configure the control file to find RID fields. The utility reads the COBOL copybook and the data files that you specify in the control file to find likely RID fields and the data values that they contain.

Alternatively, if you know the location of the RID field, you can specify it in the `fieldOffset` element in the control file. The `createdatamaps` utility validates the `fieldOffset` value that you provide, reads RID values from sample data, and matches record layouts with the data records.

By default, the utility defines one record and one table in the data map for each layout that the copybook defines, up to the maximum number that the `maxRedefines` element specifies. Also, for each layout that matches all of the data records that have a given RID value, the utility assigns an RID value to the record in the data map.

Alternatively, you can configure the utility to create data map records only for those layouts that match all of the data records that have a given RID value.

Example COBOL Copybooks with RID Fields

The following examples show COBOL copybooks with redefinitions and RID fields. For both of these examples, you can configure the createdatamaps utility to find the RID field, or use the field offset that you specify, and associate RID values with different record layouts.

The following example shows a COBOL copybook with a single 01 level, REDEFINE statements, and an RID field:

```
00001 * TRAIN6 EXAMPLE COBOL COPYBOOK
00002 01 MASTER_REC. COL 73-80
00003 05 ACCOUNT_NO PIC X(9). COL 73-80
00004 05 REC_TYPE PIC X. COL 73-80
00005 05 AMOUNT PIC S9(4)V99 COMP-3. COL 73-80
00006 05 BIN_NO PIC S9(8) COMP. COL 73-80
00007 05 BIN_NO-X REDEFINES BIN_NO PIC XXXX. COL 73-80
00008 05 DECIMAL_NO PIC S999. COL 73-80
00009 05 MASTER_DATE. COL 73-80
00010 10 DATE_YY PIC 9(2). COL 73-80
00011 10 DATE_MM PIC 9(2). COL 73-80
00012 10 DATE_DD PIC 9(2). COL 73-80
00013 05 MASTER_DOB REDEFINES MASTER_DATE. COL 73-80
00014 10 YYMMDD PIC XXXXXX. COL 73-80
00015 05 ACT_TYPE PIC X. COL 73-80
00016 05 OTHER_DATE. COL 73-80
00017 10 ODATE_YY PIC 9(2). COL 73-80
00018 10 ODATE_MM PIC 9(2). COL 73-80
00019 10 ODATE_DD PIC 9(2). COL 73-80
00020 05 OTHER_DOB REDEFINES OTHER_DATE. COL 73-80
00021 10 OYYMMDDTT PIC 9(8). COL 73-80
00022 05 OTHER_TYPE PIC X. COL 73-80
```

The createdatamaps utility can identify the RID field, REC_TYPE. Also, for each record layout that matches all of the data records that have a given RID value, the utility can assign an RID value to the record in the data map.

The following example shows a COBOL copybook with multiple 01 levels, where each 01 level defines a record type and has an RID field:

```
* train3.cob, fixed length records 60 bytes long
01 NAME_REC.
04 ACCOUNT PIC 9(3).
04 RECTYPE PIC 9(2).
04 NAME PIC X(20).
04 SEX PIC X.
04 ITEMCT PIC 9.
04 ITEMS OCCURS 3 DEPENDING ON ITEMCT PIC X(10).
04 FILLER PIC XXX.
01 ACCOUNT_REC.
04 ACCOUNT PIC 9(3).
04 RECTYPE PIC 9(2).
04 AMOUNT PIC 9(9)V99.
04 POLICY_DATE PIC X(8).
04 FILLER PIC X(36).
```

The createdatamaps utility can identify the RID field, RECTYPE, for each record type and associate the record type with an RID value.

Requirements and Limitations for Finding RID Fields

When you use the createdatamaps utility to find RID fields, the following requirements and limitations apply:

- The utility determines the field or fields that are most likely to be RID fields and associates the RID values with the record layouts that are the best match. Be sure to open the data map in the PowerExchange Navigator and confirm or edit the results.
- The utility can find RID fields only for sequential or VSAM data sources on z/OS.

- Both the data and the metadata must reside on the z/OS machine. When you start the utility from the command line, include the -pwxLocation parameter to specify the location of the PowerExchange Listener.
- The same user ID and password are required to access both data and metadata. When you start the utility from the command line, include the -pwxUserName and -pwxPassword parameters.
- A copybook can define variable-length arrays and groups. For example, a copybook might include the following line:

```
05  ARRAY OCCURS 3 DEPENDING ON ITEMCT PIC X(5) .
```

The following limitations apply to variable-length arrays and groups:

- Nested variable-length arrays or groups are not supported.
- If the metadata is variable length, the createdatamaps utility assumes that the record format is variable (RECFM=V).
- An RID field cannot follow a variable-length array or group.

Control File Elements for Finding Record IDs

You can include the following elements in the control file to configure finding RID fields:

cacheConfig

Controls the data cache on disk. You can define this element at a global level but not at a data map instance level.

The CacheConfig element includes the following elements:

cachePath

The full path to the folder for temporary working files. The cache path is written to the message log.

flushDataMode

Specifies when to flush the cache of the data records that were downloaded from the z/OS system.

Valid values:

- e - Flush the cache when the createdatamaps utility finishes.
- d - Flush the cache after each data map is created.

The default value of "e" allows data to be shared by multiple data map generations during one createdatamaps session.

excludeUnmatchedRecords

If true, generates a data map record only for those layouts for which the utility finds a valid data record ID.

findRecordIds

Enables or disables finding RIDs. You can specify findRecordIds at the global or data map instance level.

maxRedefines

Maximum number of record layouts to generate from redefinitions. You can specify maxRedefines at the global or data map instance level.

ridConfig

Defines parameters for finding RID fields. It includes following elements:

readRecordLimit

Maximum number of data records to read from each data file.

recordTypeLimit

Maximum number of record types in the data file.

fieldWidth

Maximum width in bytes of an RID field.

fieldOffset

Byte offset of the RID field from the start of the record, beginning at offset 0. If not specified, the utility finds the RID field.

matchFieldWidth

If true, the RID field must exactly match the fieldWidth value. If false, fieldWidth represents the maximum field width.

RELATED TOPICS:

- [“Schema File Reference” on page 31](#)
- [“Control File Structure” on page 26](#)
- [“How the createdatamaps Utility Determines RID Fields” on page 52](#)

How the createdatamaps Utility Determines RID Fields

When you configure the control file to find RID fields, the utility performs the following steps:

1. Identifies candidate RID fields in the metadata model. Alternatively, if you define the fieldOffset element, validates the value that you provide.
2. Obtains the candidate RID values from sample data records.
3. Matches the possible metadata layouts against sample data records, and sets the RID field and values when a good match is achieved.

If the copybook includes multiple 01 levels, the utility performs these steps for each 01 level.

Step 1 - Identifying Candidate RID Fields

The createdatamaps utility examines the COBOL copybook metadata to find candidate RID fields. The utility then determines the offset and length of the candidate RID fields.

Alternatively, if you define the fieldOffset element, the utility validates the value that you provide.

The utility uses the following principles to find candidate RID fields:

- The utility must identify at least one candidate RID field from the metadata records. If the utility finds no candidate RID fields, it stops processing the copybook for RID fields.
- RID fields must be at the same offset and have the same width for all record layouts based on a copybook.

The following element in the control file configures Step 1 processing:

- ridConfig.fieldWidth. Maximum number of bytes of an RID field.

Step 2 - Reading RID Values from Sample Data

The createdatamaps utility reads data records from the data file that is specified in the seqFileName or vsamFileName element in the control file. The utility uses the field lengths and offsets of each candidate RID field from Step 1 to read the values of the candidate RID fields in the data file. Alternatively, if you define the fieldOffset element, the utility uses the fieldOffset value to find RID values in the data file.

The utility rejects candidate RID fields based on certain checks. For example, the number of data values might exceed the maximum number of record types.

The utility uses the following principles:

- The output of Step 2 must be one RID field selected from the candidates. If more than one candidate field conforms to all the checks, the utility selects the first candidate field.
- If no candidate RID fields remain after Step 2, the utility stops processing the copybook for RID fields.
- Candidate fields are rejected if any of the following conditions apply:
 - The list of discovered RID values for the field exceeds the limit.
 - Two data records of different lengths have the same RID value.

The following elements in the control file are used in Step 2 processing:

- `ridConfig.recordTypeLimit`. Maximum number of distinct values in a valid RID field.
- `ridConfig.readRecordLimit`. Maximum number of data records to read from each data source.
- `seqFileName` or `vsamFileName`. Full path and file name of the sequential or VSAM data set that the utility reads.

Step 3 - Matching Record Layouts Against Data Records

The utility internally generates all possible record layouts, up to the number of layouts that the `maxRedefines` element specifies. The utility matches each possible layout against data records.

For each record layout, the utility checks the following conditions:

- The record length matches the length of at least one data record.
- Each field in the layout can possibly describe the data.

For each record layout that meets both conditions, the utility creates a record and a table for the layout and assigns an RID value to the record.

For each record layout that does not meet both conditions, the utility performs one of the following actions, depending on how you define the `excludeUnmatchedRecords` element:

- If `excludeUnmatchedRecords = true`, the utility excludes the layout from the data map.
- If `excludeUnmatchedRecords = false` or is not defined, the utility creates a record and a table for the layout but does not assign an RID value to the record.

The utility uses the following principles:

- This step is successful if at least one matching record layout exists for each data record type, that is, for each known RID value. This result is not guaranteed if the number of generated record layouts, limited by `maxRedefines`, is less than the number of data record types.
- Multiple record layouts might match a data record. All of these matching layouts are included in the data map. You can open the data map in the PowerExchange Navigator to view and select the correct record.

The following elements in the control file configures Step 3 processing:

- `CopybookParserConfig.maxRedefines`. The maximum number of redefines is the upper limit on the number of generated record layouts, and therefore the maximum number of record layouts to match against data records.
- `excludeUnmatchedRecords`. If true, generates a data map record only for those layouts for which the utility finds a valid data record ID.

Cache Operation

If you configure the control file to find RID fields, the utility connects to the PowerExchange Listener on the z/OS system and reads data records. The utility saves downloaded data records to a temporary disk cache to process them. The utility deletes the cache files when it finishes execution or after it generates each data map, depending on the value of the `flushDataMode` element that you specify in the control file.

The default read record limit is 10,000 records. The utility stores up to this number of records in memory. Any records above this limit are spilled to disk. For example, if the control file specifies `readRecordLimit=15,000`, 10,000 records are cached in memory and 5,000 records are spilled to disk.

The maximum record length that PowerExchange supports is 144 KB. This record length correlates to approximately 1.37GB of RAM (10000*144*1024 bytes). Ensure that the `infacmd JVM` is configured to run with a suitably large heap size. A setting of `-Xmx1500m` is usually sufficient.

To configure cache operation, define the `cacheConfig` element in the control file. The `cacheConfig` element includes the following elements:

- `cachePath`
- `flushDataMode`

For more information about `cacheConfig`, see [“CacheConfig Complex Type” on page 37](#).

Redefine Filler Fields

Because a `REDEFINE` statement defines the same section of memory in multiple ways, redefine fields and groups must be the same length as each other. If the copybook does not define these fields to be the same length, the `createdatamaps` utility inserts `FILLER` fields in the appropriate locations.

An exception to this rule is a `REDEFINE` statement that is the last item in a copybook. In this case, the redefine fields or groups can be different lengths because they typically describe different data record types, which can be different lengths.

Copybook and DBD Metadata for IMS Data Maps

For each IMS data map instance, you must specify a DBD. You can optionally specify a COBOL copybook overlay for each segment that the DBD defines. You can specify a different copybook for each segment or the same copybook for multiple segments.

Within an `imsDatamapInstance` element, specify the following elements:

- To define a DBD only, include an `importDBDDetails` element within the `imsDatamapInstance` element.
For an example, see [“Example: IMS DBD Import with No COBOL Overlay” on page 65](#)
- To define a DBD with copybook overlays, include an `importDBDDetails` element within the `imsDatamapInstance` element, and include an `OverlayDetails` element for each segment.

For an example, see [“Example: IMS DBD Import with COBOL Overlays” on page 67](#).

When you import a DBD with multiple segments, by default, the `createdatamaps` utility creates a record and a table for each segment and also creates a complex table that includes the columns from each segment in the hierarchy. To disable the creation of complex tables, specify `false` for the `createTablesForHierPath` element.

The `MaxRedefines` element is not supported for IMS data maps. For IMS data maps, the utility always selects the first `redefine`.

Column Name Filters

You can filter columns in data maps based on column name. To apply column name filters, include the `filterColumnGroup` and filter elements in the control file.

The following statements show the elements with default values:

```
<filterColumnGroup>
  <filter>
    <columnName>FILLER*</columnName>
    <exclude>true</exclude>
  </filter>
</filterColumnGroup>
```

You can include one or more filter elements in a `filterColumnGroup` element. A filter element includes the following elements:

- `columnName`. Column name to be filtered.
- `exclude`. Whether to include or exclude column names that match the criteria.
- `tableName`. Name of the table to which the criteria apply. If the `tableName` element is not specified, the filter criteria apply to all tables.

To include all columns, set `exclude` to `FALSE`:

```
<filterColumnGroup>
  <filter>
    <exclude>FALSE</exclude>
  </filter>
</filterColumnGroup>
```

The following rules apply:

- The `columnName` and `tableName` elements accept asterisks (*) and question marks (?) as wildcard characters. An asterisk represents one or more matching characters. A question mark represents a single matching character. This behavior matches that of the PowerExchange Navigator.
- If the `tableName` element is not specified, the filter condition applies to all tables.
- If multiple filter elements are defined for a table, the filter stops at the first matching criteria.
- If no `tableName` element is specified and multiple filter elements are defined, only the first filter element is considered.
- If the `exclude` element is set to `FALSE`, all columns that do not match the filter criteria are filtered.

Examples

The following example filters `FILLER*` from `TAB1`, `BLANK*` from `TAB2`, and `FILTER*` from `TAB3`. `CLEAR*` from `TAB1` is ignored, as the `FILLER*` filter is already applied to `TAB1`.

```
<filterColumnGroup>
  <filter>
    <columnName>FILLER*</columnName>
    <exclude>true</exclude>
    <tableName>TAB1</tableName>
  </filter>
  <filter>
    <columnName>BLANK*</columnName>
    <exclude>true</exclude>
    <tableName>TAB2</tableName>
  </filter>
  <filter>
    <columnName>FILTER*</columnName>
    <exclude>true</exclude>
    <tableName>TAB3</tableName>
  </filter>
</filterColumnGroup>
```

```

    </filter>
  <filter>
    <columnName>CLEAR*</columnName>
    <exclude>true</exclude>
    <tableName>TAB1</tableName>
  </filter>
</filterColumnGroup>

```

The following example filters FILLER* from all tables, as tableName is not specified. All subsequent filters are ignored.

```

<filterColumnGroup>
  <filter>
    <columnName>FILLER*</columnName>
    <exclude>true</exclude>
  </filter>
  <filter>
    <columnName>BLANK*</columnName>
    <exclude>true</exclude>
    <tableName>TAB2</tableName>
  </filter>
  <filter>
    <columnName>FILTER*</columnName>
    <exclude>true</exclude>
    <tableName>TAB3</tableName>
  </filter>
  <filter>
    <columnName>CLEAR*</columnName>
    <exclude>true</exclude>
    <tableName>TAB1</tableName>
  </filter>
</filterColumnGroup>

```

The following example filters all of the columns except BLANK* from TAB2.

```

<filterColumnGroup>

  <filter>
    <columnName>BLANK*</columnName>
    <exclude>false</exclude>
    <tableName>TAB2</tableName>
  </filter>
</filterColumnGroup>

```

Unavailable Data Map Properties

The createdatamaps utility does not provide the ability to define certain data map properties. To change the defaults for these properties, you must edit the data map in the PowerExchange Navigator.

For SEQ data maps, you cannot define the following properties in the control file:

- Fixed
- Variable
- Default
- Size
- Field Separator
- Merge Adjacent Separators
- Field Delimiter
- Encoding

- Codepage
- File List Processing

For VSAM data maps, you cannot define the following properties in the control file:

- CI ACCESS
- Data Codepage
- Number of Data Buffers
- Number of Index Buffers
- Prefix record with RRN value
- Prefix record with XRBA value
- File List Processing

For IMS data maps, you cannot define the following properties in the control file:

- Data Codepage

The defaults for these properties are described in the *PowerExchange Navigator User Guide*.

Examples

The following examples show how to create data maps for sequential, VSAM, and IMS data sources.

Control files for most of the examples are installed in the following directories on the machines where the Informatica Client and the Informatica services are installed:

- *Informatica_client_installation_directory*\clients\DeveloperClient\osgi_mf_plugins\jars\resources\examples
- *Informatica_services_installation_directory*\pwxmfplugins\resources\examples

The following control files are provided:

- ims_advanced.xml
- ims_simple.xml
- seq_advanced.xml
- seq_simple.xml
- vsam_advanced.xml
- vsam_simple.xml

To use the example control files without editing them, you must run the `createdatamaps` command from the directory on the Informatica Client or Informatica services machine in which they are installed. If you run the command from a different location, you must replace all the metadata file names in the control files with fully qualified file names before you run the command.

The following examples run the `createdatamaps` command from the Informatica services machine.

Example: Simple SEQ Data Map

This example describes a control file that creates a data map for a sequential data source and imports simple copybook metadata.

The control file defines the following properties:

- Global properties: schema name
- Data map instance properties: copybook location

Default values for the following elements are defined in the schema file:

- seqFileName = file.dat
- datamapName = MAP
- maxRedefines = 1

The copybook includes REDEFINES statements that define six possible layouts. Because maxRedefines = 1, a record and table are created for the first combination only: BIN-NO, MASTER-DATE. For more information, see the following topics:

- [“Example: SEQ Data Map with Multiple Records and Tables” on page 59](#)
- [“Redefinitions Without Record IDs in COBOL Copybooks” on page 49](#)

Command Line

To run this example on the Informatica services machine, enter the following command at the command line:

```
Informatica_services_installation_directory\isp\bin\infacmd pwx createDatamaps
-datamapOutputDir Output -controlFile seq_simple.xml -logFile Output\seq_simple.log
-verbosity INFO
```

Control File

The control file for this example, seq_simple.xml, contains the following lines:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
NOTE: Metadata file paths in this sample control file are relative to current directory.
If 'infacmd' command is issued from a different directory, all relative file paths must
be replaced with absolute file paths.
-->

<!-- xmlSchemaVersion set to 1.0 -->
<DatamapGeneration xmlSchemaVersion="1.0" xmlns="http://
com.informatica.cmd.pwx.createdatamaps/DatamapGeneration">
  <seqGen>

    <!-- Global settings for datamap file name and contents -->
    <globalGenConfig>
      <schemaName>SEQSIMPLE</schemaName>
    </globalGenConfig>

    <datamapInstances>
      <!-- Import from a copybook with default properties -->
      <seqDatamapInstance>
        <importCopybookDetails>
          <filePath>
            <windowsPath>metadata\train61.cob</windowsPath>
          </filePath>
        </importCopybookDetails>
      </seqDatamapInstance>
    </datamapInstances>
  </seqGen>
</DatamapGeneration>
```

COBOL Copybook File

The COBOL copybook for the data map in this example, tran61.cob, contains the following lines:

```
00001 * TRAIN6 EXAMPLE COBOL COPYBOOK
00002 01 MASTER_REC. COL 73-80
00003 05 ACCOUNT_NO PIC X(9). COL 73-80
00004 05 REC_TYPE PIC X. COL 73-80
00005 05 AMOUNT PIC S9(4)V99 COMP-3. COL 73-80
00006 05 BIN-NO PIC S9(8) COMP. COL 73-80
00007 05 BIN-NO-X REDEFINES BIN-NO PIC XXXX. COL 73-80
00008 05 BIN-NO-9 REDEFINES BIN-NO PIC 9(4). COL 73-80
00009 05 DECIMAL-NO PIC S999. COL 73-80
00010 05 MASTER-DATE. COL 73-80
00011 10 DATE-YY PIC 9(2). COL 73-80
00012 10 DATE-MM PIC 9(2). COL 73-80
00013 10 DATE-DD PIC 9(2). COL 73-80
00014 05 OTHER-DATE REDEFINES MASTER-DATE. COL 73-80
00015 10 OTHER-YY PIC 9(2). COL 73-80
00016 10 OTHER-MM PIC 9(2). COL 73-80
00017 10 OTHER-DD PIC 9(2). COL 73-80
```

Log File

The log file this example, seq_simple.log, contains the following lines:

```
2013-12-05 15:29:41 INFO [MDO_34613] Configuration for this run: location=, user name=,
datamap directory=Output, control file=seq_simple.xml
2013-12-05 15:29:49 INFO [MDAdapter_34100] Finding metadata. Path filter = file.dat
2013-12-05 15:29:49 INFO [MDAdapter_34100] Finding metadata. Path filter = metadata
\train61.cob
2013-12-05 15:29:49 INFO [MDAdapter 34101] Fetching file metadata\train61.cob
2013-12-05 15:29:50 INFO [MDO_34612] Copybook 'MASTER_REC' has 6 possible layouts
(Maximum configured limit is 1).
2013-12-05 15:29:50 INFO [JDMX2_34801] 1 records imported.
2013-12-05 15:29:50 INFO [JDMX2_34802] 9 fields imported.
2013-12-05 15:29:50 INFO [JDMX2_34803] 1 tables imported.
2013-12-05 15:29:50 INFO [MDO_34619] Datamap file 'Output\SEQSIMPLE.MAP.dmp' was written.
2013-12-05 15:29:55 INFO [MDO_34614] Run complete: 1 datamap(s) created. 0 error and 0
warning messages.
```

Data Map Files

This example creates a data map with the following file name and relative path:

- Output\SEQSIMPLE.MAP.dmp

Example: SEQ Data Map with Multiple Records and Tables

This example describes a control file that creates two SEQ data maps. The imported copybook metadata includes REDEFINES statements, and the data maps that are created contain multiple records and tables.

The example contains global elements and two DatamapInstance elements. At the global level, maxRedefines is set to 2. The second data map instance overrides this setting with a value of 6.

Both data maps import metadata from train61.dat, which includes the following redefinitions:

- BIN-NO-X and BIN-NO-9 both redefine BIN-NO.
- OTHER-DATE redefines MASTER-DATE.

These redefinitions result in six combinations of fields. Because the global MaxRedefines setting of 2 is in effect for the first data map, records are created for only the first two combinations:

- BIN-NO, MASTER-DATE
- BIN-NO, OTHER-DATE

Because the MaxRedefines setting of 6 is in effect for the second data map, records are created for all six combinations:

- BIN-NO, MASTER-DATE
- BIN-NO, OTHER-DATE
- BIN-NO-X, MASTER-DATE
- BIN-NO-X, OTHER-DATE
- BIN-NO-9, MASTER-DATE
- BIN-NO-9, OTHER-DATE

For more information, see ["Redefinitions Without Record IDs in COBOL Copybooks" on page 49](#).

Command Line

To run this example on the Informatica services machine, enter the following command at the command line:

```
Informatica_services_installation_directory\isp\bin\infacmd pwx createDatamaps
-datamapOutputDir Output -controlFile seq_advanced.xml -logFile Output\seq_advanced.log
-verbosity INFO
```

Control File

The control file for this example, seq_advanced.xml, contains the following lines:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
NOTE: Metadata file paths in this sample control file are relative to current directory.
If 'infacmd' command is issued from a different directory, all relative file paths must
be replaced with absolute file paths.
-->

<!-- xmlSchemaVersion set to 1.0 -->
<DatamapGeneration xmlSchemaVersion="1.0" xmlns="http://
com.informatica.cmd.pwx.createdatamaps/DatamapGeneration">
  <seqGen>

    <!-- Global settings for copybooks -->
    <globalCopybookParserConfig>
      <startColumn>7</startColumn>
      <endColumn>72</endColumn>
      <maxRedefines>2</maxRedefines>
    </globalCopybookParserConfig>

    <!-- Global settings for datamap file name and contents -->
    <globalGenConfig>
      <schemaName>SEQADV</schemaName>
      <datamapName>TRAIN6</datamapName>
    </globalGenConfig>

    <datamapInstances>

      <!-- Datamap 1: maxRedefines is 2 from global settings, resulting in 2
      datamap records -->
      <seqDatamapInstance>
        <importCopybookDetails>
          <filePath>
            <windowsPath>metadata\train61.cob</windowsPath>
          </filePath>
        </importCopybookDetails>
        <datamapProperties>
          <seqFileName>
            <zPath>COM.INFA.SEQ1</zPath>
          </seqFileName>
        </datamapProperties>
      </seqDatamapInstance>

      <!-- Datamap 2: maxRedefines value overridden to 6, resulting in 6 datamap
      records -->
```

```

    <seqDatamapInstance>
      <genConfig>
        <datamapName>TRN6REDEF</datamapName>
      </genConfig>
      <importCopybookDetails>
        <filePath>
          <windowsPath>metadata\train61.cob</windowsPath>
        </filePath>
        <parserConfig>
          <maxRedefines>6</maxRedefines>
        </parserConfig>
      </importCopybookDetails>
      <datamapProperties>
        <seqFileName>
          <zosPath>COM.INFA.SEQ2</zosPath>
        </seqFileName>
      </datamapProperties>
    </seqDatamapInstance>
  </datamapInstances>

</seqGen>
</DatamapGeneration>

```

COBOL Copybook File

The COBOL copybook for the data map in this example, tran61.cob, contains the following lines:

```

00001 * TRAIN6 EXAMPLE COBOL COPYBOOK
00002 01 MASTER_REC. COL 73-80
00003 05 ACCOUNT_NO PIC X(9). COL 73-80
00004 05 REC_TYPE PIC X. COL 73-80
00004 05 AMOUNT PIC S9(4)V99 COMP-3. COL 73-80
00005 05 BIN-NO PIC S9(8) COMP. COL 73-80
00006 05 BIN-NO-X REDEFINES BIN-NO PIC XXXX. COL 73-80
00006 05 BIN-NO-9 REDEFINES BIN-NO PIC 9(4). COL 73-80
00007 05 DECIMAL-NO PIC S999. COL 73-80
00008 05 MASTER-DATE. COL 73-80
00009 10 DATE-YY PIC 9(2). COL 73-80
00010 10 DATE-MM PIC 9(2). COL 73-80
00011 10 DATE-DD PIC 9(2). COL 73-80
00012 05 OTHER-DATE REDEFINES MASTER-DATE. COL 73-80
00013 10 OTHER-YY PIC 9(2). COL 73-80
00014 10 OTHER-MM PIC 9(2). COL 73-80
00015 10 OTHER-DD PIC 9(2). COL 73-80

```

Log File

The log file for this example, seq_advanced.log, contains the following lines:

```

2013-12-05 15:29:57 INFO [MDO_34613] Configuration for this run: location=, user name=,
datamap directory=Output, control file=seq_advanced.xml
2013-12-05 15:30:05 INFO [MDAdapter_34100] Finding metadata. Path filter = COM.INFA.SEQ1
2013-12-05 15:30:05 INFO [MDAdapter_34100] Finding metadata. Path filter = metadata
\train61.cob
2013-12-05 15:30:05 INFO [MDAdapter_34101] Fetching file metadata\train61.cob
2013-12-05 15:30:05 INFO [MDO_34612] Copybook 'MASTER_REC' has 6 possible layouts
(Maximum configured limit is 2).
2013-12-05 15:30:06 INFO [JDMX2_34801] 2 records imported.
2013-12-05 15:30:06 INFO [JDMX2_34802] 18 fields imported.
2013-12-05 15:30:06 INFO [JDMX2_34803] 2 tables imported.
2013-12-05 15:30:06 INFO [MDO_34619] Datamap file 'Output\SEQADV.TRAIN6.dmp' was written.
2013-12-05 15:30:06 INFO [MDAdapter_34100] Finding metadata. Path filter = COM.INFA.SEQ2
2013-12-05 15:30:06 INFO [MDAdapter_34100] Finding metadata. Path filter = metadata
\train61.cob
2013-12-05 15:30:06 INFO [MDAdapter_34101] Fetching file metadata\train61.cob
2013-12-05 15:30:06 INFO [MDO_34612] Copybook 'MASTER_REC' has 6 possible layouts
(Maximum configured limit is 6).
2013-12-05 15:30:06 INFO [JDMX2_34801] 6 records imported.
2013-12-05 15:30:06 INFO [JDMX2_34802] 54 fields imported.
2013-12-05 15:30:06 INFO [JDMX2_34803] 6 tables imported.

```

```
2013-12-05 15:30:06 INFO [MDO_34619] Datamap file 'Output\SEQADV.TRN6REDEF.dmp' was
written.
2013-12-05 15:30:12 INFO [MDO_34614] Run complete: 2 datamap(s) created. 0 error and 0
warning messages.
```

Data Map Files

This example creates data maps with the following file names and relative paths:

- Output\SEQADV.TRAIN6.MAP.dmp
- Output\SEQADV.TRN6REDEF.dmp

The schema name for each data map is taken from the global settings. The data map name for the first and second data maps is taken from the global setting and the data map instance setting, respectively.

Example: Simple VSAM KSDS Data Map

This example describes a control file that creates a KSDS data map and imports simple copybook metadata.

The control file is similar to the one for creating a simple SEQ data map. The control file defines the following properties:

- Global properties: schema name
- Data map instance properties: copybook location

Default values for all other properties are defined in the schema file. Because KSDS is the default for globalMapType, this example creates a KSDS data map.

Command Line

To run this example on the Informatica services machine, enter the following command at the command line:

```
Informatica_services_installation_directory\isp\bin\infacmd pwx createDatamaps
-datamapOutputDir Output -controlFile vsam_simple.xml -logFile Output\vsam_simple.log
-verbosity INFO
```

Control File

The control file for this example, vsam_simple.xml, contains the following lines:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
NOTE: Metadata file paths in this sample control file are relative to current directory.
If 'infacmd' command is issued from a different directory, all relative file paths must
be replaced with absolute file paths.
-->

<!-- xmlSchemaVersion set to 1.0 -->
<DatamapGeneration xmlSchemaVersion="1.0" xmlns="http://
com.informatica.cmd.pwx.createdatamaps/DatamapGeneration">
  <vsamGen>

    <!-- Global settings for datamap file name and contents -->
    <globalGenConfig>
      <schemaName>VSAMSIMPLE</schemaName>
    </globalGenConfig>

    <datamapInstances>

      <!-- Import from a copybook with default properties -->
      <vsamDatamapInstance>
        <importCopybookDetails>
          <filePath>
            <windowsPath>metadata\train61.cob</windowsPath>
          </filePath>
        </importCopybookDetails>
```

```

        </vsamDatamapInstance>
    </datamapInstances>

    </vsamGen>
</DatamapGeneration>

```

COBOL Copybook

This example imports metadata from the train61.cob copybook. For the contents of this copybook, see [“Example: Simple SEQ Data Map” on page 58](#).

Log File

The log file for this example, vsam_simple.log, contains the following lines:

```

2013-12-05 15:30:14 INFO [MDO_34613] Configuration for this run: location=, user name=,
datamap directory=Output, control file=vsam_simple.xml
2013-12-05 15:30:21 INFO [MDAdapter_34100] Finding metadata. Path filter = file.dat
2013-12-05 15:30:21 INFO [MDAdapter_34100] Finding metadata. Path filter = metadata
\train61.cob
2013-12-05 15:30:21 INFO [MDAdapter_34101] Fetching file metadata\train61.cob
2013-12-05 15:30:22 INFO [MDO_34612] Copybook 'MASTER_REC' has 6 possible layouts
(Maximum configured limit is 1).
2013-12-05 15:30:22 INFO [JDMX2_34801] 1 records imported.
2013-12-05 15:30:22 INFO [JDMX2_34802] 9 fields imported.
2013-12-05 15:30:22 INFO [JDMX2_34803] 1 tables imported.
2013-12-05 15:30:22 INFO [MDO_34619] Datamap file 'Output\VSAMSIMPLE.MAP.dmp' was
written.
2013-12-05 15:30:28 INFO [MDO_34614] Run complete: 1 datamap(s) created. 0 error and 0
warning messages.

```

Data Map File

This example creates a data map with the following file name and relative path:

- Output\VSAMSIMPLE.MAP.dmp

Example: VSAM RRDS Data Maps with Multiple Records and Tables

This example describes a control file that creates two VSAM RRDS data maps. The imported copybook metadata includes REDEFINES statements, and the data maps that are created contain multiple records and tables.

This example is similar to the one for creating SEQ data maps with multiple records and tables. For more information about how the REDEFINES statements in the COBOL copybook result in multiple records and tables, see [“Example: SEQ Data Map with Multiple Records and Tables” on page 59](#).

Because the control file sets the value of GlobalMapType to RRDS, the example creates RRDS data maps.

Command Line

To run this example on the Informatica services machine, enter the following command at the command line:

```

Informatica_services_installation_directory\isp\bin\infacmd pwx createDatamaps -
datamapOutputDir Output -controlFile vsam_advanced.xml -logFile Output\vsam_advanced.log
-verbosity INFO

```

Control File

The control file for this example, vsam_advanced.xml, contains the following lines:

```

<?xml version="1.0" encoding="UTF-8"?>
<!--
NOTE: Metadata file paths in this sample control file are relative to current directory.
If 'infacmd' command is issued from a different directory, all relative file paths must

```

```

be replaced with absolute file paths.
-->

<!-- xmlSchemaVersion set to 1.0 -->
<DatamapGeneration xmlSchemaVersion="1.0" xmlns="http://
com.informatica.cmd.pwx.createdatamaps/DatamapGeneration">
  <vsamGen>

    <!-- Global settings for copybooks -->
    <globalCopybookParserConfig>
      <startColumn>7</startColumn>
      <endColumn>72</endColumn>
      <maxRedefines>2</maxRedefines>
    </globalCopybookParserConfig>

    <!-- Global settings for datamap file name and contents -->
    <globalGenConfig>
      <schemaName>VSAMADV</schemaName>
      <datamapName>TRAIN6</datamapName>
    </globalGenConfig>

    <!-- Global access method for VSAM datamaps -->
    <globalMapType>RRDS</globalMapType>

    <datamapInstances>

      <!-- Datamap 1: maxRedefines is 2 from global settings, resulting in 2
datamap records -->
      <vsamDatamapInstance>
        <importCopybookDetails>
          <filePath>
            <windowsPath>metadata\train61.cob</windowsPath>
          </filePath>
        </importCopybookDetails>
        <datamapProperties>
          <vsamFileName>
            <zsoPath>COM.INFA.RRDS1</zsoPath>
          </vsamFileName>
        </datamapProperties>
      </vsamDatamapInstance>

      <!-- Datamap 2: maxRedefines value overridden to 6, resulting in 6 datamap
records -->
      <vsamDatamapInstance>
        <genConfig>
          <datamapName>TRN6REDEF</datamapName>
        </genConfig>
        <importCopybookDetails>
          <filePath>
            <windowsPath>metadata\train61.cob</windowsPath>
          </filePath>
          <parserConfig>
            <maxRedefines>6</maxRedefines>
          </parserConfig>
        </importCopybookDetails>
        <datamapProperties>
          <vsamFileName>
            <zsoPath>COM.INFA.RRDS2</zsoPath>
          </vsamFileName>
        </datamapProperties>
      </vsamDatamapInstance>
    </datamapInstances>

  </vsamGen>
</DatamapGeneration>

```

COBOL Copybook

This example imports metadata from the train61.cob copybook. For the contents of this copybook, see [“Example: Simple SEQ Data Map” on page 58](#).

Log File

The log file for this example, `vsam_advanced.log`, contains the following lines:

```
2013-12-05 15:30:30 INFO [MDO_34613] Configuration for this run: location=, user name=,
datamap directory=Output, control file=vsam_advanced.xml
2013-12-05 15:30:38 INFO [MDAdapter_34100] Finding metadata. Path filter = COM.INFA.RRDS1
2013-12-05 15:30:38 INFO [MDAdapter_34100] Finding metadata. Path filter = metadata
\train61.cob
2013-12-05 15:30:38 INFO [MDAdapter_34101] Fetching file metadata\train61.cob
2013-12-05 15:30:38 INFO [MDO_34612] Copybook 'MASTER_REC' has 6 possible layouts
(Maximum configured limit is 2).
2013-12-05 15:30:39 INFO [JDMX2_34801] 2 records imported.
2013-12-05 15:30:39 INFO [JDMX2_34802] 18 fields imported.
2013-12-05 15:30:39 INFO [JDMX2_34803] 2 tables imported.
2013-12-05 15:30:39 INFO [MDO_34619] Datamap file 'Output\VSAMADV.TRAIN6.dmp' was
written.
2013-12-05 15:30:39 INFO [MDAdapter_34100] Finding metadata. Path filter = COM.INFA.RRDS2
2013-12-05 15:30:39 INFO [MDAdapter_34100] Finding metadata. Path filter = metadata
\train61.cob
2013-12-05 15:30:39 INFO [MDAdapter_34101] Fetching file metadata\train61.cob
2013-12-05 15:30:39 INFO [MDO_34612] Copybook 'MASTER_REC' has 6 possible layouts
(Maximum configured limit is 6).
2013-12-05 15:30:39 INFO [JDMX2_34801] 6 records imported.
2013-12-05 15:30:39 INFO [JDMX2_34802] 54 fields imported.
2013-12-05 15:30:39 INFO [JDMX2_34803] 6 tables imported.
2013-12-05 15:30:39 INFO [MDO_34619] Datamap file 'Output\VSAMADV.TRN6REDEF.dmp' was
written.
2013-12-05 15:30:44 INFO [MDO_34614] Run complete: 2 datamap(s) created. 0 error and 0
warning messages.
```

Data Map Files

This example creates data maps with the following file names and relative paths:

- `Output\VSAMADV.TRAIN6.dmp`
- `Output\VSAMADV.TRN6REDEF.dmp`

Example: IMS DBD Import with No COBOL Overlay

This example describes a control file that creates an IMS data map and imports DBD metadata. This example does not import COBOL copybook metadata to overlay each segment.

The `createdatamaps` utility creates a record and a table for each of the two segments defined in the DBD. The utility also creates a complex table that includes columns from the STUDENT parent record and the CORSECN child record.

Although the DBD redefines the CRSEKEY field in the CORSECN record, the utility generates a table and record for the first redefine only. For IMS data maps, `maxRedefines` always equals 1. The log file reports which redefine was used and which redefines were skipped.

Command Line

To run this example on the Informatica services machine, enter the following command at the command line:

```
Informatica_services_installation_directory\isp\bin\infacmd pwx createDatamaps
-datamapOutputDir Output -controlFile ims_simple.xml -logFile Output\ims_simple.log
-verbosity INFO
```

Control File

The control file for this example, `ims_simple.xml`, contains the following lines:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
NOTE: Metadata file paths in this sample control file are relative to current directory.
```

If 'infacmd' command is issued from a different directory, all relative file paths must be replaced with absolute file paths.

```
-->

<!-- xmlSchemaVersion set to 1.0 -->
<DatamapGeneration xmlSchemaVersion="1.0" xmlns="http://
com.informatica.cmd.pwx.createdatamaps/DatamapGeneration">
  <imsGen>

    <!-- Global settings for datamap file name and contents -->
    <globalGenConfig>
      <schemaName>IMSSIMPLE</schemaName>
    </globalGenConfig>

    <datamapInstances>
      <!-- Import from a DBD with default properties -->
      <imsDatamapInstance>
        <importDBDDetails>
          <filePath>
            <windowsPath>metadata\train8.dbd</windowsPath>
          </filePath>
        </importDBDDetails>
      </imsDatamapInstance>
    </datamapInstances>

  </imsGen>
</DatamapGeneration>
```

DBD File

The DBD file used in this example, train8.dbd, contains the following lines:

```
DBD      NAME=DTLSTDNT, ACCESS= (HIDAM, VSAM)
DATASET DD1=DTLSTDNT
SEGM
  NAME=STUDENT, PARENT=0, FREQ=10000, BYTES=210, PTR=TB
  LCHILD NAME= (STUDIDX, DTLSTDIX), PTR=INDX
  FIELD TYPE=C, START=162, BYTES=12, NAME= (ID, SEQ, U)
  FIELD TYPE=C, START=01, BYTES=40, NAME=PNAME
  FIELD TYPE=C, START=41, BYTES=40, NAME=ADDRESS1
  FIELD TYPE=C, START=81, BYTES=40, NAME=ADDRESS2
  FIELD TYPE=C, START=121, BYTES=30, NAME=CITY
  FIELD TYPE=C, START=151, BYTES=2, NAME=STATE
  FIELD TYPE=C, START=153, BYTES=9, NAME=ZIP
  FIELD TYPE=C, START=174, BYTES=6, NAME=BDATE
  FIELD TYPE=C, START=180, BYTES=1, NAME=SEX
  FIELD TYPE=C, START=181, BYTES=2, NAME=HEIGHT
  FIELD TYPE=C, START=183, BYTES=3, NAME=WEIGHT
  FIELD TYPE=C, START=186, BYTES=5, NAME=HAIR
  FIELD TYPE=C, START=191, BYTES=5, NAME=EYES
  FIELD TYPE=C, START=196, BYTES=4, NAME=ENRLMMYY
  FIELD TYPE=C, START=200, BYTES=4, NAME=GRADMMYY
*
  SEGM NAME=CORSECTN, PARENT= ( (STUDENT, SNGL) ), FREQ=05, BYTES=14, PTR=TB
  FIELD TYPE=C, START=01, BYTES=14, NAME= (CRSEKEY, SEQ, U)
  FIELD TYPE=C, START=01, BYTES=08, NAME=CRSCOURS
  FIELD TYPE=C, START=09, BYTES=01, NAME=CRSSECTN
  FIELD TYPE=C, START=10, BYTES=01, NAME=CRSDAY
  FIELD TYPE=C, START=11, BYTES=04, NAME=CRSBEG
DBDGEN
FINISH
END
```

Log File

The log file for this example, ims_simple.log, contains the following lines:

```
2013-12-05 15:30:46 INFO [MDO_34613] Configuration for this run: location=, user name=,
datamap directory=Output, control file=ims_simple.xml
2013-12-05 15:30:54 INFO [MDAdapter_34100] Finding metadata. Path filter = metadata
```

```

\train8.dbd
2013-12-05 15:30:54 INFO [MDAdapter_34101] Fetching file metadata\train8.dbd
2013-12-05 15:30:55 INFO [MDAdapter_34108] Definition CRSEKEY selected
2013-12-05 15:30:55 INFO [MDAdapter_34109] Redefinition CRSCOURS skipped
2013-12-05 15:30:55 INFO [MDAdapter_34109] Redefinition CRSSECTN skipped
2013-12-05 15:30:55 INFO [MDAdapter_34109] Redefinition CRSDAY skipped
2013-12-05 15:30:55 INFO [MDAdapter_34109] Redefinition CRSBEG skipped
2013-12-05 15:30:55 INFO [JDMX2_34801] 2 records imported.
2013-12-05 15:30:55 INFO [JDMX2_34802] 16 fields imported.
2013-12-05 15:30:55 INFO [JDMX2_34803] 3 tables imported.
2013-12-05 15:30:55 INFO [MDO_34619] Datamap file 'Output\IMSSIMPLE.MAP.dmp' was written.
2013-12-05 15:31:01 INFO [MDO_34614] Run complete: 1 datamap(s) created. 0 error and 0
warning messages.

```

Data Map File

This example creates a data map with the following file name and relative path:

- Output\IMSSIMPLE.MAP.dmp

Example: IMS DBD Import with COBOL Overlays

This example describes a control file that creates two IMS data maps. The first data map imports DBD metadata only. The second data map imports DBD metadata and the overlaying COBOL copybook metadata for each segment.

The records in the resulting data map derive the fields and CCKs from the COBOL copybooks but retain the search fields from the DBD.

To define the DBDs and COBOL copybooks to import, the second `imsDatamapInstance` element includes the following elements:

- A `importDBDDetails` element defines the file name and path of the DBD.
- For each of the two segments defined in the DBD, an `overlayDetails` element defines the file path of the COBOL copybook and the name of the segment for which the copybook provides overlaying metadata.

The second `imsDatamapInstance` element also includes a `datamapProperties` element that defines the data map type, IMS SSID, PSB name, and PCB name.

As in [“Example: IMS DBD Import with No COBOL Overlay” on page 65](#), although the DBD redefines the CRSEKEY field in the CORSECN record, the utility generates a table and record for the first redefine only.

Command Line

To run this example on the Informatica services machine, enter the following command at the command line:

```

Informatica_services_installation_directory\isp\bin\infacmd pwx createDatamaps
-datamapOutputDir Output -controlFile ims_advanced.xml -logFile Output\ims_advanced.log
-verbosity INFO

```

Control File

The control file for this example, `ims_advanced.xml`, contains the following lines:

```

<?xml version="1.0" encoding="UTF-8"?>
<!--
NOTE: Metadata file paths in this sample control file are relative to current directory.
If 'infacmd' command is issued from a different directory, all relative file paths must
be replaced with absolute file paths.
-->

<!-- xmlSchemaVersion set to 1.0 -->
<DatamapGeneration xmlSchemaVersion="1.0" xmlns="http://
com.informatica.cmd.pwx.createdatamaps/DatamapGeneration">
  <imsGen>

```

```

<!-- Global settings for copybooks -->
<globalCopybookParserConfig>
  <startColumn>7</startColumn>
  <endColumn>72</endColumn>
</globalCopybookParserConfig>

<!-- Global settings for datamap file name and contents -->
<globalGenConfig>
  <schemaName>IMSADV</schemaName>
  <datamapName>TRAIN8</datamapName>
</globalGenConfig>

<datamapInstances>

  <!-- Datamap 1: Import from a DBD without segment overlays -->
  <imsDatamapInstance>

    <importDBDDetails>
      <filePath>
        <windowsPath>metadata\train8.dbd</windowsPath>
      </filePath>
    </importDBDDetails>

    <datamapProperties>
      <mapType>DL1</mapType>
      <imsSSID>SS1</imsSSID>
      <pcbNumber>1</pcbNumber>
    </datamapProperties>
  </imsDatamapInstance>

  <!-- Datamap 2: Import from a DBD with both segments overlaid -->
  <imsDatamapInstance>
    <genConfig>
      <datamapName>TRAIN8OVR</datamapName>
    </genConfig>

    <importDBDDetails>
      <filePath>
        <windowsPath>metadata\train8.dbd</windowsPath>
      </filePath>
    </importDBDDetails>

    <!-- Overlay segment 'STUDENT' with a Cobol copybook -->
    <overlayDetails>
      <nativeRecordName>STUDENT</nativeRecordName>
      <overlayCopybookDetails>
        <filePath>
          <windowsPath>metadata\student.cob</windowsPath>
        </filePath>
      </overlayCopybookDetails>
    </overlayDetails>

    <!-- Overlay segment 'CORSECTN' with a Cobol copybook -->
    <overlayDetails>
      <nativeRecordName>CORSECTN</nativeRecordName>
      <overlayCopybookDetails>
        <filePath>
          <windowsPath>metadata\course.cob</windowsPath>
        </filePath>
      </overlayCopybookDetails>
    </overlayDetails>

    <datamapProperties>
      <mapType>ODBA</mapType>
      <imsSSID>SS1</imsSSID>
      <psbName>psb</psbName>
      <pcbName>pcb</pcbName>
    </datamapProperties>
  </imsDatamapInstance>
</datamapInstances>

```

```

        </imsDatamapInstance>
    </datamapInstances>
</imsGen>
</DatamapGeneration>

```

DBD File

This example uses the train8.dbd file. For the contents of this file, see [“Example: IMS DBD Import with No COBOL Overlay” on page 65.](#)

COBOL Copybook Files

The following lines show the contents of the student.cob copybook. This copybook overlays the DBD metadata for the first segment in the second data map.

```

*****
*
*   COBOL FD DEFINITION FOR STUDENT FILE
*
*****
01  STUDENT-RECORD.
    04  ST-NAME                                PIC X(040) .
    04  ST-ADDRESS-1                          PIC X(040) .
    04  ST-ADDRESS-2                          PIC X(040) .
    04  ST-CITY                               PIC X(030) .
    04  ST-STATE                             PIC X(002) .
    04  ST-ZIP                               PIC X(009) .
    04  ST-NUMBER                            PIC 9(012) .
    04  ST-BIRTH-DATE.
        08  ST-BIRTH-MM                      PIC 9(002) .
        08  ST-BIRTH-DD                      PIC 9(002) .
        08  ST-BIRTH-YY                      PIC 9(002) .
    04  ST-SEX                               PIC X(001) .
    04  ST-HEIGHT                            PIC 9(002) .
    04  ST-WEIGHT                            PIC 9(003) .
    04  ST-HAIR                              PIC X(005) .
    04  ST-EYES                              PIC X(005) .
    04  ST-DATE-ENROLL-MM                    PIC 9(002) .
    04  ST-DATE-ENROLL-YY                    PIC 9(002) .
    04  ST-DATE-GRAD-MM                      PIC 9(002) .
    04  ST-DATE-GRAD-YY                      PIC 9(002) .
    04  ST-TUITION-FEES                      PIC S9(8) COMP.
    04  ST-COURSE-COUNT                      PIC X(003) .
    04  ST-COURSE-DATA OCCURS 10.
        08  ST-COURSE-CODE                    PIC 9(005) .
        08  ST-COURSE-HOURS                    PIC 9(002) .
        08  ST-COURSE-TIME                     PIC X(005) .
        08  ST-COURSE-DAY                     PIC X(005) .
        08  ST-COURSE-INSTRUCTOR              PIC X(015) .
        08  ST-COURSE-BLDG                     PIC 9(002) .

```

The following lines show the contents of the course.cob copybook. This copybook overlays the DBD metadata for the second segment in the second data map.

```

*****
*
*   COBOL FD DEFINITION FOR COURSE
*
*****
01  COURSE.
    04  CRS-COURSE                          PIC X(8) .
    04  CRS-SECTN                           PIC 9(1) .
    04  CRS-DAY                             PIC 9(1) .
    04  CRS-BEG                             PIC X(4) .
    04  CRS-END                             PIC X(4) .
    04  FILLER                             PIC X(4) .

```

Log File

The log file for this example, `ims_advanced.log`, contains the following lines:

```
2013-12-05 15:31:03 INFO [MDO_34613] Configuration for this run: location=, user name=,
datamap directory=Output, control file=ims_advanced.xml
2013-12-05 15:31:10 INFO [MDAdapter_34100] Finding metadata. Path filter = metadata
\train8.dbd
2013-12-05 15:31:10 INFO [MDAdapter_34101] Fetching file metadata\train8.dbd
2013-12-05 15:31:11 INFO [MDAdapter_34108] Definition CRSEKEY selected
2013-12-05 15:31:11 INFO [MDAdapter_34109] Redefinition CRSCOURS skipped
2013-12-05 15:31:11 INFO [MDAdapter_34109] Redefinition CRSSECTN skipped
2013-12-05 15:31:11 INFO [MDAdapter_34109] Redefinition CRSDAY skipped
2013-12-05 15:31:11 INFO [MDAdapter_34109] Redefinition CRSBEG skipped
2013-12-05 15:31:11 INFO [JDMX2_34801] 2 records imported.
2013-12-05 15:31:11 INFO [JDMX2_34802] 16 fields imported.
2013-12-05 15:31:11 INFO [JDMX2_34803] 3 tables imported.
2013-12-05 15:31:11 INFO [MDO_34619] Datamap file 'Output\IMSADV.TRAIN8.dmp' was written.
2013-12-05 15:31:11 INFO [MDAdapter_34100] Finding metadata. Path filter = metadata
\train8.dbd
2013-12-05 15:31:11 INFO [MDAdapter_34101] Fetching file metadata\train8.dbd
2013-12-05 15:31:11 INFO [MDAdapter_34100] Finding metadata. Path filter = metadata
\student.cob
2013-12-05 15:31:11 INFO [MDAdapter_34101] Fetching file metadata\student.cob
2013-12-05 15:31:11 INFO [MDAdapter_34100] Finding metadata. Path filter = metadata
\course.cob
2013-12-05 15:31:11 INFO [MDAdapter_34101] Fetching file metadata\course.cob
2013-12-05 15:31:11 INFO [MDAdapter_34108] Definition CRSEKEY selected
2013-12-05 15:31:11 INFO [MDAdapter_34109] Redefinition CRSCOURS skipped
2013-12-05 15:31:11 INFO [MDAdapter_34109] Redefinition CRSSECTN skipped
2013-12-05 15:31:11 INFO [MDAdapter_34109] Redefinition CRSDAY skipped
2013-12-05 15:31:11 INFO [MDAdapter_34109] Redefinition CRSBEG skipped
2013-12-05 15:31:11 INFO [JDMX2_34801] 2 records imported.
2013-12-05 15:31:11 INFO [JDMX2_34802] 35 fields imported.
2013-12-05 15:31:11 INFO [JDMX2_34803] 3 tables imported.
2013-12-05 15:31:11 INFO [MDO_34619] Datamap file 'Output\IMSADV.TRAIN8OVR.dmp' was
written.
2013-12-05 15:31:17 INFO [MDO_34614] Run complete: 2 datamap(s) created. 0 error and 0
warning messages.
```

Data Map Files

This example creates data maps with the following file names and relative paths:

- `Output\IMSADV.TRAIN8.dmp`
- `Output\IMSADV.TRAIN8OVR.dmp`

Example: Finding RID Fields

This example describes a control file that is configured to find RID fields.

The control file includes the following global properties that control finding RIDs:

- `cacheConfig` (`cachePath`, `flushDataMode`)

The control file includes the following data map instance properties that control RID processing:

- `findRecordIds`
- `ridConfig` (`readRecordLimit`, `recordTypeLimit`, `fieldWidth`)

`ridConfig` settings are based on the following assumptions about the data:

- `readRecordLimit` is set to 5000 because all record types are expected to occur within the first 5000 records of the data set.
- `recordTypeLimit` is set to 2 because there are no more than two distinct record types in any data set matching this copybook.

- `fieldWidth` is set to 2 because RID values for all data records are two bytes wide.

Alternatively, you could specify the `findRecordIds` and `ridConfig` properties at the global level instead of the data map instance level.

Command Line

To run this example on the Informatica services machine, enter the following command at the command line:

```
Informatica_services_installation_directory\isp\bin\infacmd pwx createDatamaps -
pwxLocation pwx_location -datamapOutputDir Output -controlFile seq_rid.xml -logFile
Output\seq_rid.log -verbosity INFO
```

For the `-pwxLocation` parameter, specify the location of the PowerExchange Listener as specified in a `NODE` statement in the PowerExchange DBMOVER configuration file.

Control File

The control file for this example, `seq_rid.xml`, contains the following lines:

```
<?xml version="1.0" encoding="UTF-8"?>

<DatamapGeneration xmlns="http://
com.informatica.cmd.pwx.createdatamaps/DatamapGeneration">
  <seqGen>

    <!-- New for 10.0. Data cache configuration. Global setting only. -->
    <cacheConfig>
      <cachePath>c:\temp\imgcache</cachePath>
      <flushDataMode>e</flushDataMode>
    </cacheConfig>

    <globalGenConfig>
      <schemaName>SEQRID</schemaName>
      <datamapName>MAP</datamapName>
    </globalGenConfig>

    <datamapInstances>

      <seqDatamapInstance>
        <genConfig>
          <!-- New for 10.0. Flag triggers new behaviour. -->
          <findRecordIds>true</findRecordIds>
        </genConfig>

        <importCopybookDetails>
          <filePath>
            <zosPath>DEV.IMG.COBOL(RID1)</zosPath>
          </filePath>
        </importCopybookDetails>

        <datamapProperties>
          <seqFileName>
            <zosPath>DEV.IMG.DATA(RID1)</zosPath>
          </seqFileName>

          <!-- New for 10.0. Skip records and record ID configuration -->
          <skipRecordCount>0</skipRecordCount>

          <ridConfig>
            <readRecordLimit>5000</readRecordLimit>
            <recordTypeLimit>2</recordTypeLimit>
            <fieldWidth>2</fieldWidth>
          </ridConfig>
        </datamapProperties>
      </seqDatamapInstance>
    </datamapInstances>
  </seqGen>
</DatamapGeneration>
```

COBOL Copybook File

The COBOL copybook for the data map in this example, DEV.IMG.COBOL(RID1), contains the following lines:

```
01 NAME_REC.
   04 ACCOUNT                PIC 9(3).
   04 RECTYPE                PIC X(2).
   04 NAME                   PIC X(20).
   04 SEX                    PIC X.
   04 ITEMCT                 PIC 9.
   04 ITEMS OCCURS 3 DEPENDING ON ITEMCT PIC X(10).
   04 FILLER                  PIC X.
01 ACCOUNT_REC.
   04 ACCOUNT                PIC 9(3).
   04 RECTYPE                PIC X(2).
   04 AMOUNT                 PIC 9(9)V99.
   04 POLICY_DATE            PIC X(8).
   04 FILLER                  PIC X.
```

The copybook includes two 01-level records and no REDEFINE statements. The copybook defines two record layouts: one for each 01-level record.

Data File

The data file in this example, DEV.IMG.DATA(RID1), contains the following lines:

```
00501Mark Jones      M3apple    orange    pear      .
005020000012345087-12-31.
01001Shirley Wong    Flraspberry .
0100200000000025657-07-04.
02001John Jackson    M2pansy    daisy     .
03001Donald Leary     M0.
030020000000004566-01-12.
04001David Wu         M1fox      .
05001Jean Connor     F3dog      cat       rabbit    .
0500200000000091196-02-29.
06001Ronald Rose      M2horse    pony      .
060020000100000001-01-01.
07001Betsy Martin     Flwolf     .
070020000064000141-12-07.
```

The first three characters of each record in this file are the ACCOUNT field. The next two characters of each record are the RECTYPE field.

Identifying RID Fields

For each 01-level record, the createdatamaps utility goes through a three-step process to identify the RID field and associate its values with specific record layouts.

For the **NAME_REC** record, the utility goes through the following steps:

Step 1 examines the metadata model to find candidate RID fields.

The following table shows the results of Step 1 for the NAME_REC record:

Field in NAME_REC Record	Candidate RID Field?
ACCOUNT	No - field exceeds maximum width
REC_TYPE	Yes
NAME	No - field exceeds maximum width
SEX	Yes

Field in NAME_REC Record	Candidate RID Field?
ITEMCT	Yes
ITEMS	No - field exceeds maximum width, RID field cannot be a variable-length field

Based on these findings, the createdatamaps utility determines the offset and length of candidate RID fields.

Step 2 reads data records from the DEV.IMG.DATA(RID1) data set that is specified in the control file. Using the offsets and lengths of the candidate fields from Step 1, Step 2 saves the data values for each candidate RID field. The utility compares the number of distinct data values with the value of recordTypeLimit in the control file. Because recordTypeLimit=2, the utility rejects candidate fields with more than two distinct data values in the data file.

The result must be one RID field selected from the candidates. In this example, the REC_TYPE and SEX fields both meet the recordTypeLimit=2 criteria and remain valid candidate fields. The utility chooses the first candidate field, REC_TYPE, as the RID field.

Step 3 generates all possible record layouts. Because the COBOL copybook in this example includes no REDEFINE statements, Step 3 generates only one record layout for the ACCOUNT_REC record type.

After completing Step 3 for the ACCOUNT_REC record type, the utility repeats the three steps for the **NAME_REC** record type.

Step 1 examines the metadata model to find candidate RID fields.

The following table shows the results of Step 1 for the ACCOUNT_REC record:

Field in ACCOUNT_REC record	Candidate RID Field?
ACCOUNT	No - field exceeds maximum width
REC_TYPE	Yes
AMOUNT	No - field exceeds maximum width
POLICY_DATE	No - field exceeds maximum width

Step 2 reads data records from the DEV.IMG.DATA(RID1) dat set. The utility compares the number of distinct data values with the value of recordTypeLimit (2, in this example) in the control file. The result must be one RID field selected from the candidates. In this example, the REC_TYPE field is the only candidate from Step 1 and also meets the Step 2 criteria.

Step 3 generates all possible record layouts. Because the COBOL copybook in this example includes no REDEFINE statements, the step generates only one record layout for the ACCOUNT_REC record type.

After performing the three steps for each 01-level record, the utility creates a data map with the following records:

Record	REC_TYPE Value
NAME_REC	01
ACCOUNT_REC	02

Log File

The log file this example, seq_rid.log, contains the following lines:

```
2015-10-09 12:10:41 INFO [MDO_34613] Configuration for this run: location=MyListener,
user name=, datamap directory=Output, control file=seq_rid.xml
2015-10-09 12:10:46 INFO [MDAdapter_34100] Finding metadata. Path filter =
DEV.IMG.COBOL(RID1)
2015-10-09 12:10:46 INFO [MDAdapter_34101] Fetching file DEV.IMG.COBOL(TST101)
2015-10-09 12:10:47 INFO [MDAdapter_34100] Finding metadata. Path filter =
DEV.IMG.DATA(RID1)
2015-10-09 12:10:47 INFO [MDO_34641] Metadata record 'NAME_REC' - Candidate record id
field 'RECTYPE' found.
2015-10-09 12:10:47 INFO [MDO_34641] Metadata record 'NAME_REC' - Candidate record id
field 'SEX' found.
2015-10-09 12:10:47 INFO [MDO_34641] Metadata record 'NAME_REC' - Candidate record id
field 'ITEMCT' found.
2015-10-09 12:10:48 INFO [MDO_34643] Metadata record 'NAME_REC' - Valid record id field
'RECTYPE' found from reading the data records.
2015-10-09 12:10:48 INFO [MDO_34612] Copybook 'NAME_REC' has 1 possible layouts (Maximum
configured limit is 2).
2015-10-09 12:10:48 INFO [MDO_34641] Metadata record 'ACCOUNT_REC' - Candidate record id
field 'RECTYPE 1' found.
2015-10-09 12:10:48 INFO [MDO_34643] Metadata record 'ACCOUNT_REC' - Valid record id
field 'RECTYPE 1' found from reading the data records.
2015-10-09 12:10:48 INFO [MDO_34612] Copybook 'ACCOUNT_REC' has 1 possible layouts
(Maximum configured limit is 2).
2015-10-09 12:10:48 INFO [JDMX2_34801] 2 records imported.
2015-10-09 12:10:48 INFO [JDMX2_34802] 12 fields imported.
2015-10-09 12:10:48 INFO [JDMX2_34803] 2 tables imported.
2015-10-09 12:10:48 INFO [MDO_34619] Datamap file 'Output\SEQ_RID.MAP.dmp' was written.
2015-10-09 12:10:48 INFO [MDO_34614] Run complete: 1 datamap(s) created. 0 error and 0
warning messages.
```

Data Map Files

This example creates a data map with the following file name and relative path:

- Output\SEQ_RID.MAP.dmp

CHAPTER 3

DTLCCADW - Adabas PCAT Utility

This chapter includes the following topics:

- [DTLCCADW Utility Overview, 75](#)
- [DTLCCADW Utility Functions, 75](#)

DTLCCADW Utility Overview

The PCAT utility program, DTLCCADW, is used by the Adabas ECCR process to manipulate the contents of the PCAT file. The PCAT utility is controlled by settings of the parameters passed via the PARM= on the EXEC statement. There are examples of the JCL required for each function in the PowerExchange DTLEXP library with names DTLCCADx, where x corresponds to the parameter value.

Typically, these functions are used only internally by PowerExchange. However, there may be times when manual overrides are desired, which are described below. When in doubt about usage, contact Informatica Global Customer Support.

DTLCCADW Utility Functions

The DTLCCADW utility has the following functions:

- P (Populate PCAT control file)
- R (Report on PCAT control file)
- I (Insert)
- D (Delete)
- L (Reset latest sequence number)
- V (Rebuild the PCAT control file)
- A (Add)
- S (Submit ADASEL)
- T (Submit ET record extraction)
- E (ET/BT record extraction)

P (Populate PCAT Control File) Function

Example job DTLCCADP - no other parameters are required.

This function may be used after the VSAM Control File has been initially established with its 999999999 control record, to pre-populate the PCAT file with previously-created PLOG data set names. By default, when the Adabas PowerExchange ECCR is started, only the most recent archived PLOG will be recognized. So, if there is a need to collect older captured changes, this is the function to use. The list of data set names is input through DDCARD DTLCCADF either directly as SYSIN, or in a file of 80-byte card images. It is the user's responsibility to obtain those PLOG data set names. The 999999999 PCAT control record is then updated with the highest sequence number added.

Note: Use this function only after initializing the control file, not after normal operation has begun.

R (Report on PCAT Control File) Function

Example job DTLCCADR - optionally, a second parameter of control file sequence number.

Prints to SYSOUT with a DD Name of DTLCCRPT. The optional second parameter allows you to specify a file sequence number from where the report will commence. If no second parameter is specified then the whole file is printed to SYSOUT.

Note: The following functions may be of use in case of operational PLOG difficulties, not related to the Adabas PowerExchange Change processing. For instance, if the PLOG files get out of sequence operationally, these functions will ensure that the PCAT can be reset to correct data set name sequence, as well.

I (Insert) Function

Example job DTLCCADI - requires two further parameters.

The first is a PCAT control file sequence number, which must not already exist. The second is the data set name of a PLOG to be inserted. Note - DTLCCADW does NOT check that the PLOG is in the correct chronological sequence - it is the user's responsibility to ensure this.

D (Delete) Function

Example job DTLCCADD - requires a second parameter of control file sequence number.

DTLCCADW reads the PCAT control record and deletes it. If you delete the record which was the latest to be added, you must immediately run the L function (see below) to reset the latest key value in the 999999999 control record.

L (Reset Latest Sequence Number) Function

Example job DTLCCADL - no other parameters are required.

This function re-populates the "latest sequence number added" field in the 999999999 PCAT control record. The only circumstance that this function would be necessary is if the user deletes the record which is the latest added, which would invalidate the '999999999' control record.

V (Rebuild the PCAT Control File) Function

Example job DTLCCADV - no other parameters are required.

This function can be used to delete and re-build the overall PCAT control record '999999999'.

Note: The following functions should be used only under the direction of Informatica Global Customer Support.

A (Add) Function

Example job DTLCCADA - no other parameters required.

Takes the PLOG specified by the data set name in the DDCARD DTLCCPLG and creates an entry in the PCAT file, taking the highest sequence number so far added and adding 100 to it (gaps are left in the sequence in case older PLOGs need to be inserted into the sequence later). This function is automatically invoked during the PLOG flip in the JCL executing the PLCOPY function and so should not be necessary to invoke manually, in normal operation.

S (Submit ADASEL) Function

Example job DTLCCADS - requires a second parameter of PCAT file sequence number.

DTLCCADW reads the PCAT control record specified by the sequence number and constructs an ADASEL job for the PLOG data set name recorded in the control record. It submits the job (by default, DTLSELJC), which runs the ADASEL and creates an output file, the data set name of which is recorded in the control record. This function is automatically invoked by the ECCR and so should not be necessary to invoke manually in normal operation.

T (Submit ET Record Extraction) Function

Example job DTLCCADT - requires a second parameter of PCAT file sequence number.

DTLCCADW reads the PCAT control record specified by the sequence number and constructs another DTLCCADW job for the PLOG recorded in the control record, building a data set name for the output ET file using date and time parameters. It submits the job (by default, DTLETLJC), which reads the PLOG specified in the control record and creates an output file of ET/BT records, the data set name of this file then being recorded in the control record. This function is normally invoked by the ECCR and so should not be necessary to invoke manually in normal operation.

E (ET/BT Record Extraction) Function

Example job DTLCCADE - requires a second parameter of PCAT file sequence number.

This function is in fact the same as the job which is dynamically created and submitted by the T function above - the difference being that the user has to explicitly define the data set name of the output ET/BT file in the JCL, DDNAME DTLCCETL, and the name of the archived PLOG being processed in DDNAME DTLCCPLG. The ECCR normally controls this operation and this function is only provided in case of difficulties which might require manual intervention.

CHAPTER 4

DTLCUIML - IMS Log Marker Utility

This chapter includes the following topics:

- [DTLCUIML Utility Overview, 78](#)
- [DTLCUIML Utility Parameters, 79](#)
- [DTLCUIML Utility Reports, 79](#)

DTLCUIML Utility Overview

Use the DTLCUIML utility to define a marker for the IMS log-based ECCR in the IMS system log data set (SLDS). Once the IMS log-based ECCR encounters one of the markers, it triggers a message in the PowerExchange Logger which stipulates a Restart and Sequence Token for the affected Registration Tags.

These Tokens can then be used as input for the Application Maintenance Utility (DTLUAPPL) to define the start point for an extraction.

There is no limit or restriction on the number of markers being set in the IMS SLDS. The IMS Log Record ID chosen has to be unique for the individual installation, and the number needs to be part of the input parameters for the IMS log-based ECCR.

This utility is used to write user-defined records to the IMS log.

The parameters controlling the utility are specified in the SYSIN file in the JCL.

The utility runs as a standard IMS application program. There is no need to provide a specific PSB. The utility can use any PSB as long as the first PCB in the PSB is an IOPCB. The utility uses the IMS LOG Call to write IMS log records.

This utility must run as an IMS BMP job. This ensures that the IMS Log record is written into the IMS logstream and that the associated log is read by the IMS log-based Collector. In an IMS DCI situation the DTLUAPPL utility has to be used to establish an extraction point for the changed data.

DTLCUIML Utility Parameters

Each SYSIN record contains the following parameters:

- DBDNAME. IMS DBD name.
- DBID. IMS instance (Recon Identifier).
- RECID. A value in (uppercase) hexadecimal from A0 through FF. It defines the log record type for the user-defined IMS log record, so it should be different to any other user-defined values which the site is using.

Leading spaces are ignored. Records are ignored where the first non-space characters are /* so can be used as comments.

Example:

```
//SYSIN DD *
  DBDNAME=DTLD004,DBID=IMS7,RECID=A0
  DBDNAME=DTLD006,DBID=IMS7,RECID=A0
  DBDNAME=DTLD007,DBID=IMS7,RECID=A0
/*
```

DTLCUIML Utility Reports

File SYSPRINT reports validation of the input parameters and progress in writing to the IMS log.

File DFSSTAT reports IMS activity.

Sample JCL is supplied in member IMSLOGW.

SYSPRINT: Control Report

The control report shows the following information:

- Date and time when the program started. This time is also used on each user-defined log record written to the IMS log.
- Validation messages for the SYSIN records. If any record is invalid, the run aborts and no records are written to the IMS log.
- Progress messages as the records are written to the IMS log.

Example:

```
2002-10-15 14:06:14 DTLCUIML REPORT
=====
.
Input Records Read
-----
  DBDNAME=DTLD004,DBID=IMS1,RECID=A0
  DBDNAME=DTLD006,DBID=IMS1,RECID=A0
  DBDNAME=DTLD007,DBID=IMS1,RECID=A0
3 record(s) validated from the input file
.
LOG record processing begins
-----
Processing dbname=DTLD004 dbid=IMS1 recid=A0 timestamp=20021015140614
Processing dbname=DTLD006 dbid=IMS7 recid=A0 timestamp=20021015140614
Processing dbname=DTLD007 dbid=IMS7 recid=A0 timestamp=20021015140614
.
Number of LOG calls = 3
.
Run completed successfully
```

DFSSTAT: IMS Activity Report

Counts for SYS LOG CALLS will match the number of records processed from file SYSIN. All other counts are zero.

Example:

```
//DFSSTAT STATISTICS FOR: JOB=UIMLRUN  STEP=G
-----
*** PST ACCOUNTING STATISTICS ***
SYS LOG CALLS                      3
```

User-Defined Log Records

Each user-defined log record contains 35 bytes of user data. The actual IMS log record adds the standard IMS suffix to this data.

The following table describes the user-defined log records:

Field	Start	Length	Type	Description
Length	1	2	unsigned binary	Length of user-defined log record = 35 bytes.
Zeros	3	2	unsigned binary	Always hex '0000'.
Recid	5	1	char	Record ID supplied in SYSIN parameters, such as hex 'A0'.
Dbname	6	8	char	IMS DBNAME.
Dbid	14	8	char	IMS instance (Recon Identifier).
Timestamp	22	14	char	Time when program DTLCUIML ran.

CHAPTER 5

DTLINFO - Release Information Utility

This chapter includes the following topics:

- [DTLINFO Utility Overview, 81](#)
- [Supported Operating Systems for the DTLINFO Utility, 81](#)
- [Control Statement Syntax for the DTLINFO Utility, 82](#)
- [Control Statement Parameters for the DTLINFO Utility, 82](#)
- [Running the DTLINFO Utility on i5/OS, 82](#)
- [Running the DTLINFO Utility on Linux, UNIX, and Windows, 82](#)
- [Running the DTLINFO Utility on z/OS, 83](#)
- [DTLINFO Utility on i5/OS Examples, 83](#)
- [DTLINFO Utility on Linux, UNIX, and Windows Examples, 84](#)
- [DTLINFO Utility on z/OS Examples, 84](#)

DTLINFO Utility Overview

Use the DTLINFO utility to perform the following functions:

- Display the version, release, and release level for PowerExchange or for a specific PowerExchange module.
- Verify the installation of the product, a service pack, or a hotfix. For example, use the utility to determine the maintenance level of your PowerExchange software at the request of Informatica Global Customer Support.

Supported Operating Systems for the DTLINFO Utility

The DTLINFO utility can run on the following operating systems:

- i5/OS
- Linux and UNIX

- Windows
- z/OS

Control Statement Syntax for the DTLINFO Utility

Use the following syntax:

```
DTLINFO [module_name]
```

To view the release information for the PowerExchange product do not specify the *module_name* parameter.

To view the release information for a specific PowerExchange module, use the *module_name* parameter. The module name is the name of any program included in your PowerExchange installation.

Control Statement Parameters for the DTLINFO Utility

The DTLINFO utility has the following optional parameter:

module_name

Displays the version, release, and release level for a specific PowerExchange module, such as DTLREXE.

Running the DTLINFO Utility on i5/OS

To run the DTLINFO utility on i5/OS:

- To view release information for PowerExchange, enter:

```
CALL PGM(dtllib/DTLINFO)
```

To view release information for a PowerExchange module, enter:

```
CALL PGM(dtllib/DTLINFO) parm ('module_name')
```

Running the DTLINFO Utility on Linux, UNIX, and Windows

To run the DTLINFO utility on Linux, UNIX, and Windows:

1. Navigate to the Informatica PowerExchange directory.
2. Enter the dtlinfo statement in one of the following ways:

To view release information for PowerExchange, enter:

```
dtlinfo
```

To view release information for a specific PowerExchange module, enter:

```
dtlinfo module_name
```

Running the DTLINFO Utility on z/OS

The JCL for the DTLINFO utility is located in *hlq*.RUNLIB(DTLINFO), where *hlq* is the high-level qualifier used for installing PowerExchange. The DTLINFO program is located in *hlq*.LOADLIB(DTLINFO)

You can incorporate the DTLINFO job step into a batch job, or add a job card and run the DTLINFO job separately.

To run the DTLINFO utility on z/OS:

1. Define the JCL EXEC statement for the DTLINFO program.

To view release information for the PowerExchange product, do not specify a PARM value or SYSIN DD as shown in the following syntax:

```
//BLDSTEP EXEC PGM=DTLINFO
//STEPLIB DD DISP=SHR,DSN=hlq.LOADLIB
//SYSPRINT DD SYSOUT=*
```

To view release information for a specific PowerExchange module, specify a module name as the PARM value. Also, supply the library and member name for the module by using the SYSIN DD as shown in the following sample:

```
//BLDSTEP EXEC PGM=DTLINFO,PARM=('DTLREXE')
//STEPLIB DD DISP=SHR,DSN=hlq.LOADLIB
//SYSIN DD DISP=SHR,DSN=hlq.LOADLIB(DTLREXE)
//SYSPRINT DD SYSOUT=*
```

The JCL statements are:

EXEC PGM=DTLINFO

Invokes the utility.

STEPLIB DD

Defines the PowerExchange LOAD library that contains the utility.

SYSPRINT DD

Defines the print location for the report.

2. Submit the DTLINFO job.

DTLINFO Utility on i5/OS Examples

The following are examples of the DTLINFO utility on i5/OS.

DTLINFO Utility on i5/OS - Example 1

The following command displays copyright and release build information for the PowerExchange installation:

```
CALL DTLINFO
```

Example output:

```
DTLINFO Latest Version:
Copyright - 1993-2016 Informatica LLC. All rights reserved.
See patents at https://www.informatica.com/legal/patents.html.
Segment#Revision :<//pwx/prod/v961/main/source/dtlinfod/dtlinfod.
Build            :<V961_HOTFIX4_4333731><Jan  5 2016 00:13:38>
```

DTLINFO Utility on i5/OS - Example 2

The following command displays copyright and release build information for the PowerExchange module DTLREXE:

```
CALL DTLINFO DTLREXE
```

Example output:

```
DTLINFO Embedded Version History:
Copyright - 1993-2016 Informatica LLC. All rights reserved.
See patents at https://www.informatica.com/legal/patents.html.

Segment#Revision :<//pwx/prod/v961/main/source/dtlinfo/dtlinfo.c
Build            :<V961_HOTFIX4_4333731><Jan  5 2016 00:13:52>

Segment#Revision :<//pwx/prod/v961/main/source/dtlrexe/dtlrexe.c
Build            :<V961_HOTFIX4_4263007><Nov  9 2015 12:15:52>
```

DTLINFO Utility on Linux, UNIX, and Windows Examples

The following are examples of the DTLINFO utility on Linux, UNIX, and Windows.

DTLINFO Utility on Linux, UNIX, and Windows - Example 1

The following command displays the release information for PowerExchange:

```
dtlinfo
```

DTLINFO Utility on Linux, UNIX, and Windows - Example 2

The following command displays the release information for the PowerExchange module DTLREXE:

```
dtlinfo dtlrexe.exe
```

DTLINFO Utility on z/OS Examples

The following are examples of the DTLINFO utility on z/OS.

DTLINFO Utility on z/OS - Example 1

The following JCL EXEC statement does not specify a PARM value or SYSIN DD for the DTLINFO program:

```
//BLDSTEP EXEC PGM=DTLINFO  
//STEPLIB DD DISP=SHR,DSN=hlq.LOADLIB  
//SYSPRINT DD SYSOUT=*
```

DTLINFO Utility on z/OS - Example 2

The following JCL EXEC statement specifies the PowerExchange module DTLREXE as the PARM value. Also, the following SYSIN DD provides the library and member name for the module:

```
//BLDSTEP EXEC PGM=DTLINFO,PARM=('DTLREXE')  
//STEPLIB DD DISP=SHR,DSN=hlq.LOADLIB  
//SYSIN DD DISP=SHR,DSN=hlq.LOADLIB(DTLREXE)  
//SYSPRINT DD SYSOUT=*
```

CHAPTER 6

DTLREXE - Remote Execution Utility

This chapter includes the following topics:

- [DTLREXE Utility Overview, 86](#)
- [Supported Operating Systems for the DTLREXE Utility, 86](#)
- [Control Statement Syntax for the DTLREXE Utility, 87](#)
- [Control Statement Parameters for the DTLREXE Utility, 87](#)
- [Running the DTLREXE Utility on i5/OS, 93](#)
- [Running the DTLREXE Utility on Linux and UNIX, 93](#)
- [Running the DTLREXE Utility on Windows, 94](#)
- [Running the DTLREXE Utility on z/OS, 94](#)
- [DTLREXE Utility Usage Notes, 96](#)
- [DTLREXE Utility on z/OS Example, 97](#)

DTLREXE Utility Overview

Use the DTLREXE utility to perform the following tasks:

- Ping a remote PowerExchange Listener.
- Submit a remote z/OS job.
- Delete a file from a remote system.
- Run a file on a remote system.

Supported Operating Systems for the DTLREXE Utility

The DTLREXE utility can run on the following operating systems:

- i5/OS
- Linux and UNIX

- Windows
- z/OS

Control Statement Syntax for the DTLREXE Utility

Use the following syntax for the DTLREXE utility control statements:

```

prog=delete
loc=location_node
parms=file_name
[config=dbmover_override_file]
[uid=userid]
[{pwd=password|epwd=encrypted_password}]

prog=ping
loc=location_node
[config=dbmover_override_file]
[uid=userid]
[{pwd=password|epwd=encrypted_password}]

prog=submit
loc=location_node
[config=dbmover_override_file]
[uid=userid]
[{pwd=password|epwd=encrypted_password}]
[fn="your jcl"]
[mode=(job|task),{wait|nowait|timed}]
[time=<time_in_seconds>]
[submittimeout=timeout_in_seconds]
[output=output.file]
[result=result.file]

prog=system
loc=location_node
parms=file_name
[config=dbmover_override_file]

```

Control Statement Parameters for the DTLREXE Utility

DTLREXE has the following statements:

- DELETE
- PING
- SUBMIT
- SYSTEM

DELETE Statement

Use DTLREXE DELETE to delete a file from the platform where the PowerExchange Listener is running.

DELETE has the following parameters:

loc

Required. Location of the PowerExchange Listener as defined in a NODE statement in the DBMOVER configuration file.

parms

Required. The name of the file to delete. On z/OS, if you do not enclose the name in quotes, the utility provides them.

prog

Required. Set to DELETE.

config

Optional. The DBMOVER configuration file that you want the DTLREXE utility to use, when you do not want to use the default DBMOVER file or the DBMOVER file that is set in the PWX_CONFIG environment variable. This parameter is used only when you run DTLREXE on Linux, UNIX, and Windows. This parameter is ignored on IBM i or z/OS.

{pwd|epwd}

Optional. A password or encrypted password for the specified user ID. For a location on i5/OS or z/OS, you can enter a passphrase or encrypted passphrase instead.

- **pwd.** A clear text password for the specified user, which allows access to the target. If a password contains nonalphanumeric characters, it must be enclosed in double quotation marks (""). A password cannot contain embedded double quotation marks.

For access to an i5/OS or z/OS location, you can enter a valid PowerExchange passphrase instead of password. An i5/OS passphrase can be from 9 to 31 characters in length. A z/OS passphrase can be from 9 to 128 characters in length. A passphrase can contain the following characters:

- Uppercase and lowercase letters
- The numbers 0 to 9
- Spaces
- The following special characters:

' - ; # \ , . / ! % & * () _ + { } : @ | < > ?

Note: The first character is an apostrophe.

Passphrases cannot include single quotation marks ('), double quotation marks (""), or currency symbols.

If a passphrase contains spaces, you must enclose it with double-quotation marks (""), for example, "This is a passphrase". If a passphrase contains special characters, you must enclose it with triple double-quotation characters ("""), for example, """"This passphrase contains special characters ! % & *.""". If a passphrase contains only alphanumeric characters without spaces, you can enter it without delimiters.

To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER configuration member. On z/OS, the DTLCFG DD statement in the DTLREXE JCL points to the DBMOVER member. On i5/OS, the DBMOVER member is the CFG file in the PowerExchange *datalib* library. For more information, see "SECURITY Statement" in the *PowerExchange Reference Manual*.

Note: On z/OS, a valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.

- **epwd.** An encrypted password for the specified user.

For access to an i5/OS or z/OS location, you can enter an encrypted PowerExchange passphrase instead of an encrypted password. Do not encrypt a passphrase that contains invalid characters, such as double-quotation marks, single quotation marks, or currency symbols.

Note: You can use the PowerExchange Navigator to encrypt a password or passphrase.

uid

Optional. A user ID that allows access to the specified location. If you specify a user ID, you must also enter a **pwd** or **epwd** value, but do not enter both.

For a location on a supported Linux, UNIX, or Windows system, if you have enabled PowerExchange LDAP user authentication, the user ID is the enterprise user name. For more information, see the *PowerExchange Reference Manual*.

PING Statement

Use DTLREXE PING to check basic connectivity to a PowerExchange Listener.

Make sure that the DBMOVER configuration file on the machine from which you run DTLREXE PING contains a NODE statement for the Listener node.

PING has the following parameters:

loc

Required. The location of the PowerExchange Listener, as defined in a NODE statement in the DBMOVER configuration file.

prog

Required. This value must be to PING.

config

Optional. The DBMOVER configuration file that you want the DTLREXE utility to use, when you do not want to use the default DBMOVER file or the DBMOVER file that is set in the PWX_CONFIG environment variable. This parameter is used only when you run DTLREXE on Linux, UNIX, and Windows. This parameter is ignored on IBM i or z/OS.

{pwd|epwd}

Optional. A password or encrypted password for the specified user ID. For a location on i5/OS or z/OS, you can enter a passphrase or encrypted passphrase instead.

- **pwd.** A clear text password for the specified user, which allows access to the target. If a password contains nonalphanumeric characters, it must be enclosed in double quotation marks (""). A password cannot contain embedded double quotation marks.

For access to an i5/OS or z/OS location, you can enter a valid PowerExchange passphrase instead of password. An i5/OS passphrase can be from 9 to 31 characters in length. A z/OS passphrase can be from 9 to 128 characters in length. A passphrase can contain the following characters:

- Uppercase and lowercase letters
- The numbers 0 to 9
- Spaces
- The following special characters:

' - ; # \ , . / ! % & * () _ + { } : @ | < > ?

Note: The first character is an apostrophe.

Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.

If a passphrase contains spaces, you must enclose it with double-quotation marks ("), for example, "This is a passphrase". If a passphrase contains special characters, you must enclose it with triple double-quotation characters ("""), for example, """This passphrase contains special characters ! % & *. """". If a passphrase contains only alphanumeric characters without spaces, you can enter it without delimiters.

To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER configuration member. On z/OS, the DTLCFG DD statement in the DTLREXE JCL points to the DBMOVER member. On i5/OS, the DBMOVER member is the CFG file in the PowerExchange *datalib* library. For more information, see "SECURITY Statement" in the *PowerExchange Reference Manual*.

Note: On z/OS, a valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.

- **epwd.** An encrypted password for the specified user.

For access to an i5/OS or z/OS location, you can enter an encrypted PowerExchange passphrase instead of an encrypted password. Do not encrypt a passphrase that contains invalid characters, such as double-quotation marks, single quotation marks, or currency symbols.

Note: You can use the PowerExchange Navigator to encrypt a password or passphrase.

uid

Optional. A user ID that allows access to the specified location. If you specify a user ID, you must also enter a **pwd** or **epwd** value, but do not enter both.

For a location on a supported Linux, UNIX, or Windows system, if you have enabled PowerExchange LDAP user authentication, the user ID is the enterprise user name. For more information, see the *PowerExchange Reference Manual*.

SUBMIT Statement

Use DTLREXE SUBMIT to submit a job to a remote z/OS platform or server.

You can enter SUBMIT parameters at the command line. Alternatively, you can use the **cs** parameter to point to a parameter file that contains the parameters you need.

```
dtlrexe cs=<parameter_file>
```

SUBMIT has the following parameters:

loc

Required. The location of the PowerExchange Listener, as defined in a NODE statement in the DBMOVER configuration file.

prog

Required. This value must be to SUBMIT.

config

Optional. The DBMOVER configuration file that you want the DTLREXE utility to use, when you do not want to use the default DBMOVER file or the DBMOVER file that is set in the PWX_CONFIG environment variable. This parameter is used only when you run DTLREXE on Linux, UNIX, and Windows. This parameter is ignored on IBM i or z/OS.

fn

Optional. The name of the file on z/OS that contains the JCL to be submitted, including the job name. Use the following format:

```
fn="dtlusr.jcl(yourjob)"
```

On Windows, use the following format:

```
fn="dtlusr.jcl(yourjob)\"
```

mode

Optional. Specifies the submit mode. Use the following format:

```
mode={job|task},{wait|nowait|timed}
```

The available modes are:

- **job**. A submitted job
- **task**. A started task. (Not currently supported.)
- **wait**. Synchronous. Report result at the end and wait for completion.
- **nowait**. Asynchronous. Submit the job but do not wait until report completion.
- **timed**. Synchronous. Waits for the length of time that is specified by the time parameter.

output

Optional. The file name for the file that contains the results from the job. Use the following format:

```
output=dtlusr.output
```

If the output is a PDS member, the same format requirements are in place as for the **fn** parameter.

{pwd|epwd}

Optional. A password or encrypted password for the specified user ID. For a location on i5/OS or z/OS, you can enter a passphrase or encrypted passphrase instead.

- **pwd**. A clear text password for the specified user, which allows access to the target. If a password contains nonalphanumeric characters, it must be enclosed in double quotation marks ("). A password cannot contain embedded double quotation marks.

For access to an i5/OS or z/OS location, you can enter a valid PowerExchange passphrase instead of password. An i5/OS passphrase can be from 9 to 31 characters in length. A z/OS passphrase can be from 9 to 128 characters in length. A passphrase can contain the following characters:

- Uppercase and lowercase letters
- The numbers 0 to 9
- Spaces
- The following special characters:

```
' - ; # \ , . / ! % & * ( ) _ + { } : @ | < > ?
```

Note: The first character is an apostrophe.

Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.

If a passphrase contains spaces, you must enclose it with double-quotation marks ("), for example, "This is a passphrase". If a passphrase contains special characters, you must enclose it with triple double-quotation characters ("""), for example, """This passphrase contains special characters ! % & * .""". If a passphrase contains only alphanumeric characters without spaces, you can enter it without delimiters.

To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER configuration member. On z/OS, the DTLCFG DD statement in the DTLREXE JCL points to the DBMOVER member. On i5/OS, the DBMOVER member is the CFG file in the PowerExchange *datalib* library. For more information, see "SECURITY Statement" in the *PowerExchange Reference Manual*.

Note: On z/OS, a valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.

- **epwd.** An encrypted password for the specified user.

For access to an i5/OS or z/OS location, you can enter an encrypted PowerExchange passphrase instead of an encrypted password. Do not encrypt a passphrase that contains invalid characters, such as double-quotation marks, single quotation marks, or currency symbols.

Note: You can use the PowerExchange Navigator to encrypt a password or passphrase.

result

Optional. The file to which the results from the job are written on the client platform where DTLREXE runs.

The file specification must be suitable for the platform.

If the output is a PDS member, the same format requirements are in place as for the **fn** parameter.

submittimeout

Optional. The time, in seconds, to wait for the submitted job to start running.

time

Optional. The time, in seconds, to wait for the job to return results. This wait period starts when the job is submitted.

uid

Optional. A user ID that allows access to the specified location. If you specify a user ID, you must also enter a **pwd** or **epwd** value, but do not enter both.

For a location on a supported Linux, UNIX, or Windows system, if you have enabled PowerExchange LDAP user authentication, the user ID is the enterprise user name. For more information, see the *PowerExchange Reference Manual*.

SYSTEM Statement

Use DTLREXE SYSTEM to run a program file from the path or steplib. You can specify a file such as a batch file, rexx, or executable file.

SYSTEM has the following parameters:

loc

Required. The location of the PowerExchange Listener, as defined in a NODE statement in the DBMOVER configuration file.

parms

Required. The name of the program file to run.

prog

Required. This value must be SYSTEM.

config

Optional. The DBMOVER configuration file that you want the DTLREXE utility to use, when you do not want to use the default DBMOVER file or the DBMOVER file that is set in the PWX_CONFIG environment variable. This parameter is used only when you run DTLREXE on Linux, UNIX, and Windows. This parameter is ignored on IBM i or z/OS.

Running the DTLREXE Utility on i5/OS

To run the DTLREXE utility on i5/OS:

- Enter the following command:

```
CALL PGM(DTLREXE) PARM('prog=submit loc=mvs fn=dtlusr.load.jcl mode=(job,wait)
output=dtlusr.output, result=dtlusr.result')
```

Running the DTLREXE Utility on Linux and UNIX

You can run the DTLREXE utility on Linux and UNIX specifying either a PDS member or a sequential MVS data set.

Submitting a Remote z/OS Job Specifying a PDS Member

To submit a remote z/OS job specifying a PDS member:

- Enter the following command, specifying a PDS member as follows:

```
dtlrexe prog=submit loc=remlist fn=\"dtlusr.jcl.cntl('db2load')\" ,
mode=('job,wait')', output=dtlusr.output, result=/usr/pwx/output.txt
```

Alternatively, you can enter the following command:

```
dtlrexe prog=submit cs=/usr/pwx/MyParameterFile.txt
```

Submitting a Remote z/OS Job Specifying a Sequential MVS Data Set

To submit a remote z/OS job specifying a sequential MVS data set:

- Enter the following command, specifying a sequential MVS data set as follows:

```
dtlrexe prog=submit loc=remlist fn=dtlusr.load.jcl, mode=('job,wait')',
output=dtlusr.output, result=/usr/pwx/output.txt
```

Alternatively, you can enter the following command:

```
dtlrexe prog=submit loc=remlist fn="dtlusr.load.jcl", mode=('job,wait')',
output=dtlusr.output, result=/usr/pwx/output.txt
```

Deleting a File from a Remote System

To delete a file from a remote system:

- Enter the following command:

```
dtlrexe prog=delete loc=location parms=file_name
```

Running a File on a Remote System

To run a file on a remote system:

- ▶ Enter the following command:

```
dtlrexe prog=system loc=location parms=file_name
```

For example:

```
dtlrexe prog=system loc=node1 parms=Q:\mydir\myprog.bat
```

Running the DTLREXE Utility on Windows

To run the DTLREXE utility on Windows:

- ▶ Enter the following command:

```
dtlrexe prog=submit loc=remlist fn="dtlusr.jcl.cntl(db2load)" mode=(job,nowait)  
output=dtlusr.output result=c:\submit\output\output.txt uid=user01 pwd=pass01
```

Alternatively, you can enter the following command:

```
dtlrexe prog=submit cs=c:\PowerExchange\MyParameterFile.txt
```

RELATED TOPICS:

- [“Running the DTLREXE Utility on Linux and UNIX” on page 93](#)

Running the DTLREXE Utility on z/OS

You can run the DTLREXE utility on z/OS with PROG=SUBMIT, PROG=PING, PROG=DELETE, or PROG=SYSTEM. In each case, the JCL statements are:

JOB

Initiates the job.

EXEC PGM=DTLREXE

Invokes the utility.

STEPLIB DD

Defines the PowerExchange LOAD library that contains the utility.

SYSPRINT DD

Defines the print location for the report.

Running the DTLREXE Utility with PROG=SUBMIT

To run the DTLREXE utility with PROG=SUBMIT:

1. Edit the DTLREXE job JCL. The following two lines must be the first step of the job:

```
//START EXEC PGM=DTLNTS,PARM='%STRTJOB'  
//STEPLIB DD DSN=&HLQ..LOADLIB,DISP=SHR
```

Then use the following JCL for the DTLREXE job step:

```
//STEP1 EXEC PGM=DTLREXE,
//      PARM=('CS=DD:INCMD'),
//      REGION=0M,TIME=NOLIMIT

//INCMD DD *
LOC=NODE1 PROG=SUBMIT FN="DTLUSR.JCL(MYJOB)"
MODE=(JOB,WAIT) OUTPUT=DTLUSR.DB2LOAD.SYSPRINT
RESULT="DTLUSR.JCLRESTXT)"
```

After the final step, you must add the following lines:

```
//      IF ((RC > 4) | (ABEND=TRUE)) THEN
// *
//ENDERR EXEC PGM=DTLNLS,
//      PARM="'%ENDJOB" C 16'
//STEPLIB DD DSN=&HLQ..LOADLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
// *
//      ELSE
// *
//ENDOK EXEC PGM=DTLNLS,
//      PARM="'%ENDJOB'"
//STEPLIB DD DSN=&HLQ..LOADLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//      ENDIF
```

2. Verify the JCL.
3. Submit the DTLREXE job.

Running the DTLREXE Utility with PROG=PING

To run the DTLREXE utility with PROG=PING:

1. Edit the job that you intend to submit using DTLREXE as follows:

```
//STEP1 EXEC PGM=DTLREXE,
//      PARM='loc=node1 prog=ping'
//STEPLIB DD DSN=CEE.SCEERUN,
//      DISP=SHR
//      DD DSN=&HLQ..LOADLIB,
//      DISP=(SHR)
//SYSPRINT DD SYSOUT=*
```

Enter the location of the PowerExchange Listener in the loc parameter.

2. Submit the DTLREXE job.

Running the DTLREXE Utility with PROG=DELETE

To run the DTLREXE utility with PROG=DELETE:

1. Edit the job that you intend to submit using DTLREXE as follows:

```
//STEP1 EXEC PGM=DTLREXE,
//      PARM='loc=node1 prog=delete parms=file_name'
//STEPLIB DD DSN=CEE.SCEERUN,
//      DISP=SHR
//      DD DSN=&HLQ..LOADLIB,
//      DISP=(SHR)
//SYSPRINT DD SYSOUT=*
```

Enter the location of the PowerExchange Listener in the loc parameter.

2. Submit the DTLREXE job.

Running the DTLREXE Utility with PROG=SYSTEM

To run the DTLREXE utility with PROG=SYSTEM:

1. Edit the job that you intend to submit using DTLREXE as follows:

```
//STEP1      EXEC PGM=DTLREXE,  
//           PARM='loc=node1 prog=system parms=file_name'  
//STEPLIB    DD DSN=CEE.SCEERUN,  
//           DISP=SHR  
//           DD DSN=&HLQ..LOADLIB,  
//           DISP=(SHR)  
//SYSPRINT   DD SYSOUT=*
```

Enter the location of the PowerExchange Listener in the loc parameter.

2. Submit the DTLREXE job.

DTLREXE Utility Usage Notes

Consider the following points before using the DTLREXE utility:

- DTLREXE submits the job on the host named in the loc parameter.
- If the mode is (job,nowait), the output and result data sets are of no interest.
- If the mode is (job,wait) or (job,timed), PowerExchange waits for the job to complete and reads the return code. The parameters are required to ensure that the job has completed and the output data set is available.
- Substitution is performed on the job for the %STRTJOB and %ENDJOB tokens.

The following table describes the %STRTJOB and %ENDJOB tokens:

Parameter	Description
%STRTJOB	The name token for the first step in the JCL of the job that is to be submitted. <ul style="list-style-type: none">- If the mode parameter is set to (job,wait/timed), %STRTJOB is substituted with a name token generated by the submitter.- If the mode parameter is not set to (job,wait/timed), %STRTJOB is set to DONOTRETURNOKEN.
%ENDJOB	The name token for the last step in the JCL of the job that is to be submitted. The wait/timed processing retrieves these values to determine if the job has started, is running, or has finished. The %ENDJOB steps have to be included manually and are shown in the sample JCL. If the submitted job fails with a return code greater than four, rc=16 is returned back to DTLREXE on the client.

- To print help on the utility, run DTLREXE without any parameters.

DTLREXE Utility on z/OS Example

To launch DTLREXE from a z/OS job you must use PowerExchange command set syntax as follows:

```
//STEP1 EXEC PGM=DTLREXE,  
//          PARM=('CS=DD:INCMD'),  
//          REGION=0M,TIME=NOLIMIT
```

An inline DD is specified in the JCL above. You can change this to an external member.

The specified inline or external DD contains the parameters of the DTLREXE command. The following JCL defines the inline DD:

```
//INCMD DD *  
LOC=NODE1 PROG=SUBMIT FN="DTLUSR.JCL(MYJOB) "  
MODE=(JOB, WAIT) OUTPUT=DTLUSR.DB2LOAD.SYSPRINT  
RESULT="DTLUSR.JCLRESTXT) "
```

The following JCL specifies the external member:

```
//INCMD DD DSN=HLQ..RUNLIB(MYCS)
```

The member MYCS has the following contents:

```
LOC=NODE1 PROG=SUBMIT FN="DTLUSR.JCL(MYJOB) "  
MODE=(JOB, WAIT) OUTPUT=DTLUSR.DB2LOAD.SYSPRINT  
RESULT="DTLUSR.JCLRESTXT) "
```

DTLREXE Utility on z/OS Example JCL

This example uses the following JCL:

```
//DTLREXE JOB 'DTLREX',MSGLEVEL=(1,1),MSGCLASS=X,CLASS=A,  
//          NOTIFY=&SYSUID  
// *  
//          SET HLQ=DTLUSR.V850  
// *  
//STEP1 EXEC PGM=DTLREXE,REGION=24M,  
//          PARM=('CS=DD:INCMD')  
//STEPLIB DD DSN=CEE.SCEERUN,DISP=SHR  
//          DD DSN=&HLQ..LOADLIB,DISP=SHR  
//DTLCFG DD DSN=&HLQ..RUNLIB(DBMOVER),DISP=SHR  
//DTLKEY DD DSN=&HLQ..RUNLIB(LICENSE),DISP=SHR  
//DTLMSG DD DSN=&HLQ..DTLMSG,DISP=SHR  
//DTLLOG DD DSN=&HLQ..LOG,DISP=SHR  
//SYSUDUMP DD SYSOUT=*  
//SYSOUT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//CEEDUMP DD SYSOUT=*  
//SYSIN DD *  
// * EXECUTE THE PROCEDURE  
// *  
//DTLLSTN EXEC DBMOVE  
//INCMD DD *  
LOC=NODE1 PROG=SUBMIT FN="DTLUSR.JCL(MYJOB) "  
MODE=(JOB, WAIT) OUTPUT=DTLUSR.DB2LOAD.SYSPRINT  
RESULT="DTLUSR.JCL(RESTXT) "
```

DTLREXE Utility on z/OS Output Data Set

The output parameter indicates a data set that should contain the results of the submitted job.

When the job completes, the output is read and transferred back to the client where it is written to a file specified by the `result=` parameter.

The format of the output is:

timestamp|jobid|text

An example of the output is:

```
20060223172636000000|JOB03370|1DSNU000I DSNUGUTC - OUTPUT START FOR UTILITY, UTILID =
DB2LDJCL |
20060223172636000000|JOB03370|0DSNU050I DSNUGUTC - LOAD DATA RESUME NO REPLACE LOG YES|
20060223172636000000|JOB03370| DSNU650I -DSN7 DSNURWI - INTO TABLE DTLUSR.T3|
20060223172636000000|JOB03370| DSNU650I -DSN7 DSNURWI - (COL1 POSITION(3) CHAR(100)
NULLIF(1='Y'),|
20060223172636000000|JOB03370| DSNU650I -DSN7 DSNURWI - COL2 POSITION(*) CHAR(100)
NULLIF(2='Y'))|
20060223172636000000|JOB03370| DSNU350I -DSN7 DSNURRST - EXISTING RECORDS DELETED FROM
TABLESPACE|
20060223172636000000|JOB03370| DSNU304I -DSN7 DSNURWT - (RE)LOAD PHASE STATISTICS -
NUMBER OF RECORDS=3 FOR TABLE DTLUSR.T3 |
20060223172636000000|JOB03370| DSNU302I DSNURILD - (RE)LOAD PHASE STATISTICS - NUMBER OF
INPUT RECORDS PROCESSED=3 |
20060223172636000000|JOB03370| DSNU300I DSNURILD - (RE)LOAD PHASE COMPLETE, ELAPSED
TIME=00:00:08|
20060223172636000000|JOB03370| DSNU010I DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST
RETURN CODE=0 |
```

CHAPTER 7

DTLUAPPL - Restart Token Utility

This chapter includes the following topics:

- [DTLUAPPL Utility Overview, 99](#)
- [Supported Operating Systems for the DTLUAPPL Utility, 100](#)
- [DTLUAPPL Control Statement Syntax, 100](#)
- [Connection Statement, 101](#)
- [ADD and MOD Statements, 102](#)
- [END APPL Statement, 105](#)
- [PRINT APPL Statement, 105](#)
- [Running the DTLUAPPL Utility on i5/OS, 105](#)
- [Running the DTLUAPPL Utility on Linux, UNIX, and Windows, 106](#)
- [Running the DTLUAPPL Utility on z/OS, 106](#)
- [DTLUAPPL Utility Examples, 108](#)

DTLUAPPL Utility Overview

Use the DTLUAPPL utility to generate restart tokens for CDC sessions and to add, modify, or print application information in the CDEP file. DTLUAPPL can generate restart tokens for all PowerExchange CDC source types and for CDC sessions that use either ODBC or PowerExchange Client for PowerCenter (PWXPC) connections.

Restart tokens determine the point in the change stream at which CDC sessions begin extracting change data. Restart tokens are composed of a pair of sequence and restart token values. The lengths of the token values depend on the data source type and whether you use ODBC or PWXPC.

If you use ODBC connections to extract change data, PowerExchange maintains application names, registrations associated with each application name, and the restart tokens in the CDEP file. When you use the utility to add or modify applications, the utility updates the CDEP file. If an application name entry does not exist when the initial extraction process runs, PowerExchange creates an application entry in the CDEP file. To generate current restart tokens for the initial extraction process, run the utility immediately after target materialization and before any change data has been captured or applied. After the initial extraction process runs, you can use DTLUAPPL to generate current restart tokens for an application.

If you use PWXPC connections to extract change data, the CDEP file is not used. After you materialize the target tables and before you start the CDC session, run DTLUAPPL to generate restart tokens that identify the starting point for extraction processing. Then update the PWXPC restart token file with these generated restart tokens. If you rematerialize target tables for any reason, you can run the utility again to generate current restart tokens.

Note: The utility generates sequence tokens in the format that is used by ODBC extractions. To use generated sequence tokens for a PWXPC session, use the utility to print the sequence and restart token values for the capture registrations associated with the application name. Then add eight zeroes to the end of the printed sequence values.

Alternatively, you can generate current restart tokens from the PowerExchange Navigator. For more information, see the *PowerExchange Navigator User Guide*.

Supported Operating Systems for the DTLUAPPL Utility

The DTLUAPPL utility can run on the following operating systems:

- i5/OS
- Linux and UNIX
- Windows
- z/OS

DTLUAPPL Control Statement Syntax

Use the following syntax to add or modify an application, generate restart tokens at the application level or registration level, or print information for an application:

```
UID user_ID EPWD encrypted_password [CONN_OVER capi_connection_name]
{ADD|MOD} APPL application_name instance [RSTKN GENERATE]
    [CAPTMETH=access_method]
    [CONDTYPE=P]
    [JRN=library/journal]
    [ORACOLL=collection_id] [ORACONN=connection] [ORAINST=instance]
    [ORASchema=schema]
    [UDBDB=database]

    {ADD|MOD} RSTTKN registration_name [DB=library/table] [GENERATE]
        [SEQUENCE sequence_token]
        [RESTART restart_token]
END APPL application_name
[PRINT APPL {application_name|ALL}]
```

In this syntax:

- The UID, EPWD, and CONN_OVER parameters comprise the connection statement that is used for generating restart tokens or specifying a CAPI_CONNECTION override.
- The ADD or MOD APPL statement is required to add or modify an application name. Under the ADD or MOD APPL statement, you can define optional parameters that depend on the source type and also specify one or more ADD or MOD RSTTKN statements for specifying restart tokens for registrations associated with the application. The GENERATE parameter can be specified at either the application level or registration level.
- The END APPL statement terminates the ADD or MOD APPL statement.
- The PRINT APPL statement prints information from the CDEP file for specific applications or all applications that are involved in ODBC extractions.

You can specify a combination of ADD, MOD, and PRINT statements in a single request. The following example adds an application, specifies sequence and restart tokens for three capture registrations, generates restart tokens at the application level for the remaining capture registrations, and prints information about the application:

```
ADD  APPL IMSAPP1 IMS1 rsttkn GENERATE
      add rsttkn d002long
        sequence 00000A036E160000000000000A036BAA00000000
        RESTART  AAAAAAAAA4040000000002BA700000000
      add rsttkn d002root
        SEQUENCE 00000A036E160000000000000A036BBB0000000
        RESTART  AAAAAAAAA4040000000002BA700000000
      add rsttkn d003root
      add rsttkn d008addr
      add rsttkn d008pay
      add rsttkn d008skil
        SEQUENCE 00000A036E160000000000000A036CCC00000000
        RESTART  AAAAAAAAA4040000000002BA700000000
END  IMSAPP1
PRINT APPL ALL
```

Connection Statement

The connection statement is comprised of the UID and EPWD parameters and optional CONN_OVR parameter.

The UID and EPWD parameters specify a user ID and encrypted password. Only PowerExchange-encrypted passwords are allowed. These parameters are required for generating the following types of restart tokens;

- Restart tokens for DB2 for i5/OS, Microsoft SQL Server, MySQL, Oracle, PostgreSQL, and DB2 for Linux, UNIX, and Windows capture registrations
- Restart tokens that are used when PowerExchange security for i5/OS or z/OS is enabled by the SECURITY statement in the DBMOVER configuration file

The optional CONN_OVR parameter specifies a CAPI_CONNECTION override when the default CAPI_CONNECTION in the DBMOVER configuration file is not appropriate for generating restart tokens.

Parameters:

UID *user_ID*

Specifies the system user identifier.

EPWD *encrypted_password*

Specifies the PowerExchange-encrypted password for the specified user ID. You can generate encrypted passwords in the PowerExchange Navigator.

CONN_OVR *capi_connection_name*

Specifies the name of the CAPI_CONNECTION statement that the DTLUAPPL utility uses when the default CAPI_CONNECTION statement is not suitable. Ensure that the override CAPI_CONNECTION statement is defined in the DBMOVER configuration file. If you do not specify this parameter, DTLUAPPL uses the default CAPI_CONNECTION statement.

ADD and MOD Statements

Use the ADD and MOD statements to generate or specify restart tokens for ODBC or PWXPC extractions. You can also use these statements to add or modify applications and the capture registrations associated with an application. For example, if you create a new capture registration, you can modify an existing application to include it.

For ODBC extractions, the utility updates application and registration information in the CDEP file based on the ADD and MOD statements.

Note: The information in the CDEP file does not apply to PWXPC extractions.

You can configure the ADD and MOD statement statements to generate restart tokens at the application level or registration level or to add the sequence and restart tokens that you specify for a capture registration in these statements to the CDEP file.

To generate restart tokens, use the RSTTKN GENERATE keyword at the application level or the GENERATE keyword at the registration level. Restart tokens generated at the registration level override those generated at the application level. Restart tokens generated at the application level apply to any capture registrations in ADD or MOD RSTTKN statements that do not include a GENERATE parameter or specific SEQUENCE and RESTART values.

If you use ODBC extractions, PowerExchange maintains restart tokens for each registration associated with an application in the CDEP file.

If you use PWXPC extractions, PWXPC maintains the restart tokens in the state table for a relational target database on the target or in the state file for a nonrelational target on the Integration Service machine. For more information about PWXPC and restart token management, see *PowerExchange Interfaces for PowerCenter*.

When you define ADD or MOD statements, use the following rules and guidelines:

- To add a new application, use the ADD APPL and ADD RSTTKN statements.
- To modify an existing application, use the MOD APPL statement.
- To add or modify an existing capture registration for an existing application, use the MOD RSTTKN statement.
- If you attempt to add an application that already exists, DTLUAPPL produces an error.
- The ADD or MOD APPL statement must always end with an END APPL statement.
- For PWXPC extractions, do not use the application name that is specified in the PWXPC application connection for generating restart tokens.

Parameters:

application_name

Specifies the name of the application to be added or modified. This value is case sensitive.

instance

Specifies the source instance. Ensure that this value matches the instance value that is displayed in the PowerExchange Navigator for the registration group and extraction group. The type of value varies by data source type.

The following table identifies the instance value type for each source type:

Data Source	Instance Value
Adabas	The DBID value.
Datacom	The MUF Name value that is specified for the registration group.
Db2 for i (i5/OS)	The DBID value in the CAPTPARM member of the CFG file.
Db2 on Linux, UNIX, or Windows (Db2 LUW)	The database name that is specified for the registration group.
Db2 for z/OS	The Db2 subsystem ID.
IDMS	The CV name.
IMS	The IMS system identifier.
Microsoft SQL Server	The instance value that is specified for the registration group
MySQL	The instance value that is specified for the registration group.
Oracle	The <i>collection_identifier</i> value that is specified in the ORACLEID statement in the DBMOVER configuration file.
PostgreSQL	The instance value that is specified for the registration group
VSAM	The instance value that is specified for the registration group.

Note: Values from the registration group definition are in all uppercase. Use the same case when specifying the values in this parameter.

RSTTKN GENERATE

Generates restart tokens that mark the current end of the change stream at the application level. The generated restart tokens apply to the capture registrations associated with application for which restart tokens are not generated at the registration level or specifically defined in SEQUENCE and RESTART keywords.

CAPTMETH=*access_method*

Specifies one of the following capture access methods:

- **CAPXRT**. For real-time or continuous extraction mode.
- **CAPX**. For batch extraction mode.

Valid only for the following CDC sources: Db2 LUW, Oracle, Microsoft SQL Server, MySQL, and PostgreSQL.

CONDTYPE={P|F}

Specifies the condense type for which DTLUAPPL generates restart tokens.

Enter one of the following options:

- **P**. Partial condense processing by the PowerExchange Logger for Linux, UNIX, or Windows or by PowerExchange Condense on i5/OS or z/OS. Valid for all source types.

- **F.** Full condense processing by PowerExchange Condense on i5/OS and z/OS. Valid for Datacom, DB2 for z/OS, and VSAM sources on z/OS that specify key columns. Also valid for DB2 for i5/OS tables that have primary keys and DDS files that are defined with a unique key.

No default.

JRN=library/journal

Overrides the Db2 for i journal that is specified in the capture registration.

Valid for Db2 for i CDC only.

ORACOLL=collection_id

Overrides the collection identifier that is recorded for the capture registration in the CCT file.

Valid for Oracle CDC only.

ORACONN=source_connect_string

Overrides the Oracle connection information for a specific Oracle collection ID, which is specified as the third positional parameter in the ORACLEID statement in the dbmover.cfg configuration file. By using this override in conjunction with ORAINST, you can use a single set of capture registrations to capture data from multiple Oracle instances.

You can specify ORACONN or ORAINST or both. If one of these parameter values is not specified, PowerExchange uses the parameter value from the ORACLEID statement in dbmover.cfg configuration file for the missing parameter.

Valid for Oracle CDC only.

ORAINST=instance

Overrides the Oracle instance name for a specific Oracle collection ID, which is specified as the second positional parameter in the ORACLEID statement in the dbmover.cfg configuration file. By using this override in conjunction with ORACONN, you can use a single set of capture registrations to capture data from multiple Oracle instances.

Valid for Oracle CDC only.

ORASchema=schema

Overrides the Oracle schema name that is specified for the registration group. By using this override, you can use of a single set of capture registrations to capture data from multiple schemas in an Oracle instance.

Valid for Oracle CDC only.

UDBDB=database

Specifies the connection database when it is different from the registration database.

Valid for Db2 LUW CDC only.

{ADD|MOD} RSTTKN registration_name [DB=library/table] [GENERATE]

Adds a capture registration to an application or generates restart tokens at the registration-level.

registration_name

Specifies the name of the capture registration. This value is case sensitive.

DB=library/table

Overrides the Db2 table that is specified in the capture registration.

Valid for Db2 for i CDC only.

GENERATE

Generates restart tokens that mark the current end of the change stream for the capture registration.

END APPL Statement

The END APPL statement is required to designate the end of an entire ADD or MOD APPL control statement, including the ADD or MOD RSTTKN substatements.

PRINT APPL Statement

The PRINT APPL statement is optional. It prints the restart tokens and other information that is stored in the CDEP file for one or more specific applications or for all applications that extract data by using ODBC.

You can specify the PRINT APPL statement only or include it after ADD or MOD statements that add or modify applications or generate restart tokens. The PRINT APPL statement does not require a connection statement.

To print information for multiple applications, concatenate the PRINT APPL statements or use the ALL keyword:

```
PRINT APPL application_name  
PRINT APPL application_name
```

or

```
PRINT APPL ALL
```

The utility prints the following information:

- The application name
- The registrations associated with the application, including the registration user-defined names, tag names, and sequence and restart tokens
- The restart token for the first, last, and current runs of an application in the format for ODBC extractions

Running the DTLUAPPL Utility on i5/OS

To run the utility on i5/OS, use the following command:

```
SBMJOB CMD(CALL PGM(DTLLIB/DTLUAPPL) PARM('DATALIB/CFG(TKNPARMS)')) JOB(MYJOB)  
JOB(DATALIB/DTLLIST) PRTDEV(*JOB) OUTQ(*JOB) CURLIB(DATALIB) INLLIB(*JOB)
```

The CFG/TKNPARMS member in the *datalib* library contains the utility control statements.

Running the DTLUAPPL Utility on Linux, UNIX, and Windows

To run the utility on Linux, UNIX, or Windows, use the following commands:

Command	Description
dtluappl	This command assumes the utility control statements are in the dtltknf.txt file. The command displays output in the command window. Note: The dtltknf.txt file resides in the PowerExchange installation directory, along with the DTLUAPPL program. The file contains sample control statements for the utility, which you can customize.
dtluappl > logname.txt	This command assumes the utility control statements are in the dtltknf.txt file. The command writes output to the logname.txt file.
dtluappl parm_file.txt > logname.txt	This command reads the utility control statements from a user-defined parameter file. If you create the file in a directory other than the PowerExchange base installation directory, provide the full path and file name, for example, C:\mydir\runuappl.txt. The command writes output to the logname.txt file.

Running the DTLUAPPL Utility on z/OS

PowerExchange provides sample JCL for the DTLUAPPL utility in the DTLUAPPL member of the RUNLIB library.

Customize the sample JCL, as needed, and then submit the job.

For example, your customized JCL might contain the following statements:

```
//jobname JOB
//LIBSRCH JCLLIB ORDER=your.RUNLIB
//INCS1 INCLUDE MEMBER=GENBULK
//INCS3 INCLUDE MEMBER=GENCHNG
//STEP1 EXEC PGM=DTLUAPPL
//STEPLIB DD DISP=SHR,DSN=&HLQ..LOADLIB
// DD DISP=SHR,DSN=&HLQ..LOAD
// DD DISP=SHR,DSN=&SCERUN
//EDMPARMS DD DISP=SHR,DSN=&HLQEDM..&LOGGER&SUFFIX..USERLIB
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//EDMSG DD SYSOUT=*
//***
//SYSIN DD *
MOD APPL tokens DSN9 RSTTKN GENERATE
ADD RSTTKN db2demo1
END APPL tokens
PRINT APPL tokens
//*
//*
//DTLAMCPR DD DSN=&HLQVS..CCT,
// DISP=(SHR)
//DTLCACDE DD DSN=&HLQVS..CDEP,
// DISP=(SHR)
//*
//DTLMSG DD DSN=&HLQ..DTLMSG,
// DISP=(SHR)
//DTLOUT DD SYSOUT=*
```

```
//DTLCFG DD DSN=&RUNLIB (DBMOVER) ,
//      DISP= (SHR)
//DTLKEY DD DSN=&RUNLIB (LICENSE) ,
//      DISP= (SHR)
//DTLLOG DD SYSOUT=*
//DTLLOG01 DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
```

The JCL statements are:

JOB

Defines the DTLUAPPL job card to MVS, including the job name.

EXEC PGM=DTLUAPPL

Executes the DTLUAPPL program. This PGM name must be DTLUAPPL.

STEPLIB DD

Points to the PowerExchange LOADLIB and LOAD libraries and the Language Environment (LE) common runtime library.

EDMPARMS DD

Points to the USERLIB library, which contains the EDMSDIR module options that are used to connect to the PowerExchange Agent and PowerExchange Logger for z/OS.

SYSPRINT DD

Defines a SYSOUT data set to which job output is printed.

SYSUDUMP DD

Defines a SYSOUT data set for dump data that can be used to diagnose DTLUAPPL problems.

EDMMSG DD

Defines a SYSOUT data set for messages from the PowerExchange Logger, ECCRs, PowerExchange Agent, Log Read API (LRAPI), and Log Write API (LWRAPI).

SYSIN DD

Defines the input control statements for the DTLUAPPL utility. You can specify the control statements in stream or point to a data set in which you define the control statements. The example JCL contains in-stream statements for adding an application name, specifying its restart tokens, and printing information for all application names.

DTLAMCPR DD

Points to the CCT data set, which contains the capture registrations.

DTLCACDE DD

Points to the CDEP data set, which contains information for the application names that are used for ODBC change data extraction processes, PowerExchange Navigator row tests, and some other PowerExchange processes.

DTLMSG DD

Points to the data set that contains the PowerExchange messages that can be issued during PowerExchange processing, including utility processing.

DTLOUT DD

Defines a SYSOUT data set that includes messages from the DTLUAPPL utility.

DTLCFG DD

Points to the DBMOVER configuration file for PowerExchange.

DTLKEY DD

Points to the PowerExchange LICENSE member in the RUNLIB library, which contains your PowerExchange license key.

DTLLOG DD

Defines a SYSOUT data set for logging PowerExchange messages that report the status and events of some PowerExchange processes and components.

DTLLOG01 DD

Defines a SYSOUT data set for logging PowerExchange messages that report the status and events of some PowerExchange processes and components when alternative logging is enabled.

DTLUAPPL Utility Examples

The following examples demonstrate how to use the DTLUAPPL utility to perform certain functions.

Example 1. Generating Restart Tokens at the Application Level

In this example, the DTLUAPPL utility generates restart tokens for the DB2DEM01 source registration that is associated with the application name of "tokens." The utility generates restart tokens at the application level and then prints the generated restart tokens.

The example uses the following control statements:

```
UID user1 EPWD CDFB2EE51CFC16C7
ADD APPL tokens DSN7 RSTTKN GENERATE
    ADD RSTTKN db2demo1
END APPL tokens
PRINT APPL tokens
```

The GENERATE keyword is specified in the ADD APPL statement. The generated restart tokens apply to the capture registration that is specified in the ADD RSTTKN statement because no GENERATE keyword or specific SEQUENCE and RESTART values are included in the ADD RSTTKN statement.

If you use the PowerExchange Client for PowerCenter (PWXPC), after the restart tokens are printed, you can add them to the restart token file that is specified in the application connection. The token values can then be used for extraction processing. You must add eight trailing zeroes to the sequence token value for PWXPC extractions.

Example 2. Generating Restart Tokens at the Capture Registration Level

In this example, the DTLUAPPL utility generates restart tokens at the registration level for the DB2DEM01 source registration that is associated with the application name of "tokens." The utility also prints the restart tokens.

The example uses the following control statements:

```
MOD APPL tokens DSN7
    ADD RSTTKN db2demo1 GENERATE
```

```
END APPL tokens
PRINT APPL tokens
```

The GENERATE keyword is specified in the ADD RSTTKN statement and applies only to the capture registration that is specified in that statement.

If you use the PowerExchange Client for PowerCenter (PWXPC), after the restart tokens are generated, add them to the restart token file that is specified in the application connection for the extraction.

Example 3. Generating Restart Tokens for Continuous Extraction Mode

In this example, the DTLUAPPL utility generates restart tokens at the registration level for the rrtb001 capture registration that is associated with the existing "dummy" application and FOX920 source instance. The utility overrides the default CAPI_CONNECTION statement with the CAPX CAPI_CONNECTION named CAPXORA. The utility also prints the generated restart tokens.

The example uses the following control statements:

```
UID user01 EPWD 40ABC4B0E32FD99F CONN_OVR CAPXORA
MOD APPL dummy FOX920 RSTTKN GENERATE CAPTMETH=CAPXRT CONDTYPE=P
MOD RSTTKN rrtb001
END APPL dummy
PRINT APPL dummy
```

Based on these control statements, the utility generates restart tokens in the format that is required for continuous extraction mode. The utility generates restart tokens for continuous extraction of Oracle data from PowerExchange Logger for Linux, UNIX, and Windows log files because the CAPMETH parameter value is CAPXRT, the CONDTYPE parameter value is P, and the override CAPI_CONNECTION statement type is CAPX.

Example 4. Adding an Application with Restart Tokens

In this example, the DTLUAPPL utility adds the IMSAPP1 application with capture registrations d002long, d002root, d003root, and d008addr to the CDEP file.

For the d002long and d002root registrations, the ADD RSTTKN statements specify the sequence and restart tokens. For the d008addr capture registration, the utility generates the restart tokens. For the d003root capture registration, the utility adds the registration to the CDEP file without restart tokens.

The example uses the following control statements:

```
ADD APPL IMSAPP1 IMS1
ADD RSTTKN d002long
SEQUENCE 00000A036E160000000000000A036BAA00000000
RESTART AAAAAAAAA4040000000002BA700000000
ADD RSTTKN d002root
SEQUENCE 00000A036E160000000000000A036BBB00000000
RESTART AAAAAAAAA4040000000002BA700000000
ADD RSTTKN d003root
ADD RSTTKN d008addr GENERATE
```

For ODBC extractions, the utility stores the application, associated capture registrations, and restart tokens in the CDEP file. If you use PWXPC instead of ODBC, you must manually add the capture registrations and their restart tokens to the restart token file that is specified in the application connection for the extractions.

Example 5. Adding an Application and Generating Restart Tokens on a Remote Instance

In this example, the DTLUAPPL utility adds the ORAAP3 application for the remote ORAINST1 instance. The utility also generates restart tokens at the application level, which will be used for the oraemp2 source registration.

The example uses the following control statements:

```
ADD APPL ORAAP3 ORAINST1 RSTTKN GENERATE ORACONN=OCONN ORAINST=OINST ORACOLL=OCOLL
ADD RSTTKN oraemp2
END APPL ORAAP3
```

Example 6. Modifying Restart Tokens in an Application

In this example, the DTLUAPPL utility modifies the restart tokens that are used for the d002long capture registration in the IMSAPP1 application. The sequence and restart tokens are specified in the MOD RSTTKN statement and apply to the registration only.

The example uses the following control statements:

```
MOD APPL IMSAPP1 IMS1
MOD RSTTKN d002long
SEQUENCE 000000032D450000000000000000032D4500000000
RESTART C4D6C3D340400000000032CBD00000000
END APPL IMSAPP1
```

Example 7. Modifying an Application and Adding a Registration

In this example, the DTLUAPPL utility adds the d003long capture registration with specific sequence and restart token values to the existing IMSAPP1 application.

The example uses the following control statements:

```
MOD APPL IMSAPP1 IMS1
ADD RSTTKN d003long
SEQUENCE 000000032D450000000000000000032D4500000000
RESTART C4D6C3D340400000000032CBD00000000
END APPL IMSAPP1
```

The ADD RSTTKN statement adds the capture registration and the specified SEQUENCE and RESTART tokens to the CDEP file, which is used for ODBC extractions. If you use PWXPC to extract change data, you must manually add the capture registration and sequence and restart tokens to the restart token file that is specified in the application connection for the extraction.

Example 8. Printing Information for an Application

In this example, the DTLUAPPL utility prints information from the CDEP file for a specific application.

The example uses the following control statement:

```
PRINT APPL {application_name}
```

Note: You can print information for more than one application by concatenating multiple PRINT APPL statements, each with a specific application name, or by using the ALL keyword for all applications.

The utility produces the following example output if no extractions have run for the application:

```
Application name=<DB2APPL5> Rsttkn=<2> Ainseq=<0> Preconfig=<N>
FirstTkn   =<>
LastTkn    =<>
CurrentTkn =<>
Registration name=<db2v52c.1> tag=<DB2DSN1db2v52c1>
```

```

Sequence=<000000035D50000000000000000000035D5000000000>
Restart  =<C4D6C3D34040000000035CC800000000>
Registration name=<db2tst5c.1> tag=<DB2DSN1db2tst5c1>
Sequence=<000000035D50000000000000000000035D5000000000>
Restart  =<C4D6C3D34040000000035CC800000000>

```

After an extraction has run for the application, the utility produces the following output, which includes the FirstTkn and LastTkn values:

```

Application name=<DB2APPL1> Rsttkn=<1> Ainseq=<0> Preconfig=<N>
FirstTkn  =<C4D6C3D340400000000335D000000000>
LastTkn   =<C4D6C3D3404000000003453E00000000>
CurrentTkn=<>
Registration name=<db2v52c.1> tag=<DB2DSN1db2v52c1>
Sequence=<0000000319140000000000000000031914000000000>
Restart=<4D6C3D3404000000003188C00000000>

```

The following table describes the fields in the PRINT APPL output:

Field	Description
Rsttkn	Number of RSTTKN pairs that are recorded for the application.
Ainseq	For internal use only.
Preconfig	Not used.
FirstTkn	The restart token for first successful run of the application when using ODBC.
LastTkn	The restart token for last successful run of the application when using ODBC.
CurrentTkn	The restart token for current active or last failed run of the application when using ODBC.

Note: If you use ODBC, you can also view the restart tokens that are in the printed output in the **Extract Application** dialog box of the PowerExchange Navigator.

CHAPTER 8

DTLUCBRG - Batch Registration Utility

This chapter includes the following topics:

- [DTLUCBRG Utility Overview, 112](#)
- [Supported Operating Systems for the DTLUCBRG Utility, 113](#)
- [DTLUCBRG Utility Parameters, 113](#)
- [DTLUCBRG Code Page Processing, 127](#)
- [Running the DTLUCBRG Utility, 127](#)
- [DTLUCBRG Utility Example Reports, 130](#)
- [DTLUCBRG Utility Usage Notes, 133](#)

DTLUCBRG Utility Overview

Many customers who use change data capture in production environments need to register hundreds of tables for capture. Using the PowerExchange Navigator to create and manage large numbers of registrations is not practical. The DTLUCBRG utility enables you to create and manage registrations in bulk.

This utility can perform the following tasks:

- Adds capture registrations and extraction maps. The utility creates registrations and extraction maps at specified PowerExchange Listener locations for a set of tables or data maps. You can use a mask to limit the registrations created.
- Generates before-image columns and change-indicator columns in the extraction maps as needed.
- Modifies inactive or active registrations.
- Performs a test run to report on the scope of the registrations before actually creating them.
- For Microsoft SQL Server sources, changes the status of multiple registrations in one operation and also deletes and rebuilds the associated SQL Server publications. This feature is useful if you need to reset the status of multiple registrations, or if you need to re-create registrations and regenerate publications because you made DDL changes to a large number of tables.

Note: DTLUCBRG creates all registrations with a version of 1. The utility cannot set the registration status to history and then create a subsequent version of the registration.

Supported Operating Systems for the DTLUCBRG Utility

DTLUCBRG is available on the following operating systems:

- i5/OS
- Linux
- UNIX
- Windows
- z/OS

You can create registrations on other platforms by using PowerExchange Listeners.

DTLUCBRG Utility Parameters

This section describes DTLUCBRG utility parameters.

Define the parameters at one of the following locations, depending on your operating system type:

- On IBM i (i5/OS), define parameters in the parameters file that you specify on the command line.
- On Linux, UNIX, and Windows, define parameters in the dtlucbrg.txt file. This file should be in the directory from which you run DTLUCBRG.
- On z/OS, define parameters in the SYSIN of the JCL.

Note: You are not required to specify parameters that have default values.

Parameter Descriptions:

CONDTYPE

The Condense option to use for the capture registrations. Options are:

- **FULL.** This option is available if you use PowerExchange Condense on i5/OS or z/OS. PowerExchange accumulates change data in keyed condense files. Because later changes supersede earlier changes, this condense option does not maintain transactional consistency. Also, the following limitations apply:
 - Adabas and IDMS log-based CDC data sources are not supported.
 - On z/OS, data sources must have key columns. The total length of all key columns for a source cannot exceed 250 bytes.
 - On i5/OS, source tables must have primary keys, or DDS files must be defined with a unique key.
- **PART.** This option is available if you use PowerExchange Condense on i5/OS or z/OS or the PowerExchange Logger for Linux, UNIX, or Windows. Changes in successfully committed UOWs are written to condense files or PowerExchange Logger log files in chronological order based on UOW end time. PowerExchange writes all changes for the columns of interest, not just the latest changes. This condense type maintains transactional consistency.
- **NONE.** Capture registrations are not eligible for full or partial condense processing.

For more information, see the *PowerExchange Navigator User Guide* or PowerExchange CDC guides.

CREATEBICI

Controls whether the DTLUCBRG utility generates a before-image column, change-indicator column, or both for data columns in extraction maps. This parameter applies to all extraction maps that the utility creates. If you specify multiple tables by using a table-name mask, the CREATEBICI keywords apply to all of the tables that match the mask. See [“DTLUCBRG Utility Usage Notes” on page 133](#) for more information about limitations related to this parameter.

The syntax of the CREATEBICI parameter is:

```
CREATEBICI=(COLUMNS={PKFK|ALL|FILE}, MAXCOLS={number_of_columns|500}, EXTINFO={BI|CI|BICI},  
INPUT_FN=input_file_name, OUTPUT_FN=output_file_name, ERROR_FN=error_file_name)
```

The CREATEBICI parameter contains the following keywords:

- **COLUMNS.** The columns in the extraction map for which the utility generates before-image columns, change-indicator columns, or both types of columns, as indicated by the EXTINFO keyword. Options are:
 - **PKFK.** Generate before-image and change-indicator columns for primary-key and foreign-key columns.
 - **ALL.** Generate before-image and change-indicator columns for all selected columns in the extraction map.
 - **FILE.** Generate before-image and change-indicator columns for columns that are defined in the table schema in the utility input file.

No default is provided.

Note: If you are generating before-image or change-indicator columns and the COLUMNS keyword identifies user-defined columns with expressions that invoke PowerExchange functions, ensure that the functions support BI buffering. If a function does not support BI buffering, PowerExchange does not generate BI or CI columns for the field. For more information, see [“Expression-Field Functions That Support CREATEBICI Processing” on page 121](#).

- **MAXCOLS.** Optional. The maximum number of columns for which the utility generates before-image and change-indicator columns in the extraction map. Valid values are 10 through 32000. Default is 500.
How you set this keyword depends on the COLUMNS setting. Use the following guidelines:
 - If COLUMNS=FILE, the MAXCOLS value must be equal to or greater than the number of entries in the input file.
 - If COLUMNS=ALL, the MAXCOLS value must be equal to or greater than the largest number of columns in a registered table.
 - If COLUMNS=PKFK, the MAXCOLS value must be equal to or greater than the number of primary and foreign keys in the largest registered table.
- **EXTINFO.** Optional. The type or types of metadata columns to generate in the extraction maps. Options are:
 - **BI.** Before-image columns.
 - **CI.** Change-indicator columns.
 - **BICI.** Both before-image and change-indicator columns.

Default is BICI.

Note: In some cases, the before-image or change-indicator column cannot be created. For example, the utility cannot create a before-image column for DB2 z/OS LOB columns in extraction maps. If you try to create a before-image column for DB2 for z/OS LOB columns, the request is ignored and a warning message is written to the output file.

- **INPUT_FN.** On i5/OS, Linux, Unix, and Windows, the name of the input file that contains the column names for the before-image and change-indicator columns. On z/OS, enter **Y** or **N**. If you enter **Y**, the utility uses a default partitioned data set extended (PDSE) that has been pre-allocated with a DSNTYPE of LIBRARY and a record length of 255 bytes. You can use this keyword only if the COLUMNS keyword is set to **FILE**.
- **OUTPUT_FN.** On i5/OS, Linux, Unix, and Windows, the name of the output file that contains log information about the before-image and change-indicator columns created in the extraction maps. On z/OS, enter **Y** or **N**. If you enter **Y**, the utility uses a default partitioned data set extended (PDSE) that has been pre-allocated with a DSNTYPE of LIBRARY and a record length of 255 bytes.
- **ERROR_FN.** On i5/OS, Linux, Unix, and Windows, the name of the file that lists the columns for which before-image and change-indicator columns failed to be generated in the extraction maps. On z/OS, enter **Y** or **N**. If you enter **Y**, the utility uses a default partitioned data set extended (PDSE) that has been pre-allocated with a DSNTYPE of LIBRARY and a record length of 255 bytes.

Note: If both CREATEBICI and RPTCOLS=Y are specified, the utility adds before-image and change-indicator columns to the output report with the value **Y** or **N** in each of these columns.

CRGNAME

A user-defined capture registration name. This name can be up to 13 alphanumeric characters in length and cannot begin with a number. PowerExchange uses this value as the entire registration name. Because PowerExchange does not append a unique number to it, as it does with CRGPREFIX, you can use CRGNAME to replace registrations that were lost or damaged or to regenerate registrations under their original registration names.

Do not use CRGNAME under any of the following circumstances:

- The TABLE parameter specifies a mask that contains the asterisk (*) wildcard. To use CRGNAME, the table name must be explicitly specified in the same manner as when registering the table.
- The REUSECRGNAME parameter is set to **Y**.
- The CRGPREFIX parameter is specified. You must specify CRGPREFIX or CRGNAME, but do not specify both of them.

CRGPREFIX

A prefix of one to four characters in length for the registration name. The utility appends a 4-digit sequential number to this prefix to form a unique registration name.

The registration name can have any of the following formats:

```
xnnnn
xxxxnnn
xxxxnnnn
xxxxnnnnn
```

Where:

- *x...* is the prefix value that you enter. The first *x* value must be a letter from a to z. Subsequent *x* values can include the letters a to z or the integers 0 to 9.
- *nnnn* is a sequential number that the utility generates, starting from 0001.

If the utility tries to create a registration for a table that has a table name containing invalid characters, the utility issues an error message and continues processing without creating the registration for the table.

Note: To replace a registration under its previous name, you must use the CRGNAME parameter instead of the CRGPREFIX parameter.

Do not specify both CRGPREFIX or CRGNAME.

DBTYPE

A three-character mnemonic that indicates the data source type. Options are:

- **ADA** for Adabas
- **AS4** for DB2 for i (i5/OS)
- **DB2** for DB2 for z/OS
- **DCM** for Datacom
- **IDL** for IDMS log-based
- **IMS** for IMS
- **MSS** for Microsoft SQL Server
- **MYS** for MySQL
- **ORA** for Oracle
- **PGS** for PostgreSQL
- **UDB** for DB2 for Linux, UNIX, and Windows
- **VSM** for VSAM

Note: Use **DB2** only for DB2 on z/OS. Use **AS4** or **UDB** for DB2 products on other platforms.

EPWD

An encrypted password for the user ID specified by UID.

If the utility accesses an i5/OS or z/OS location, you can enter an encrypted PowerExchange passphrase instead of an encrypted password. Do not encrypt a passphrase that contains invalid characters, such as double-quotation marks, single quotation marks, or currency symbols.

You can create an encrypted password or passphrase in the PowerExchange Navigator by selecting **File > Encrypt Password**.

Use EPWD instead of PWD if you are not allowed to store passwords in a readable format.

INSTANCE

The source instance for your registrations. The value type depends on the source database type that you specify in the DBTYPE parameter.

Based on the DBTYPE option, enter one of the following values:

- For **ADA**, enter an Adabas nucleus name.
- For **AS4**, enter a DB2 for i (i5/OS) instance value that matches the INST parameter value in the AS4J CAPI CONNECTION statement in the DBMOVER member of the *dtllib*/CFG file. If you use PowerExchange Condense, this instance value must also match the DBID parameter value in the CAPTPARM member.
- For **DB2**, enter a DB2 subsystem ID (SSID).
- For **DCM**, enter a Datacom Multi-User Facility (MUF) name.
- For **IDL**, enter an IDMS log-based CDC instance value that matches the *registration_logsid* parameter in the LOGSID statement in the DBMOVER configuration member.
- For **IMS**, enter an IMS subsystem ID that matches the *ims_ssid* parameter value in the IMSID statement in the DBMOVER configuration member.
- For **MSS**, optionally enter a unique user-defined instance identifier for the SQL Server database server and database name combination defined in the MSSOPTS DBSERVER and DBNAME parameters. Maximum length is seven characters. This identifier is incorporated into the names of the extraction

maps that the utility creates. If you use the PowerExchange Logger for Linux, UNIX, and Windows, ensure that the instance identifier matches the DBID parameter value in the Logger configuration file. If you do not enter this instance value, PowerExchange generates a unique instance identifier that is composed of all or part of the publication database name followed by a 3-digit number if a number is required to make the identifier unique.

This INSTANCE parameter is useful in migration scenarios. If you need to deploy change capture from one environment to another, such as from test to production, and you do *not* define an instance identifier, PowerExchange uses the generated instance identifier in the new environment. The generated instance identifier might be different than the one in the original source environment. To avoid having to update the extraction map names in PowerCenter workflows and edit the DBID parameter value for the PowerExchange Logger, enter an instance identifier in the INSTANCE parameter that matches the instance identifier in the original environment when creating registrations for the new environment.

Tip: In this migration scenario, ensure that the dbmover.cfg configuration files in the original environment and new environment specify unique paths in CAP_PATH and CAPT_XTRA statements.

- For **MYS**, enter a unique, user-defined instance identifier for the MySQL database server. Maximum length is seven characters. This identifier is used to identify a set of registrations for tables in the database on the MySQL server. This identifier is also incorporated into the names of the extraction maps that the utility creates. If you use the PowerExchange Logger for Linux, UNIX, and Windows, ensure that this instance identifier matches the DBID parameter value in the Logger configuration file.
- For **ORA**, enter the user-defined collection ID for the Oracle instance that matches the *collection_id* parameter in the ORACLEID statement in the PowerExchange DBMOVER configuration member.
- For **PGS**, enter a unique, user-defined instance identifier for the PostgreSQL database server and database name combination defined in the PGSOPTS DBSERVER and DBNAME parameters. Maximum length is seven characters. This identifier is incorporated into the names of the extraction maps that the utility creates. If you use the PowerExchange Logger for Linux, UNIX, and Windows, ensure that the instance identifier matches the DBID parameter value in the Logger configuration file. If you do not enter this instance value, PowerExchange generates a unique instance identifier that is composed of all or part of the database name followed by a three-digit number.
- For **UDB**, enter a DB2 for Linux, UNIX, and Windows database name.
- For **VSM**, enter a VSAM collection identifier.

LOCATION

Required. A node name that points to the location of the PowerExchange Listener that manages the capture registrations and extraction maps.

If the registrations, data maps, and data source reside on the same system, you can specify LOCATION=LOCAL. In this case, do not define the other LOCATION_xxx parameters.

Warning: Do not specify LOCATION=LOCAL if you add or modify VSAM, IMS synchronous, or DB2 for z/OS capture registrations. Otherwise, the PowerExchange Agent fails to detect the new or updated registrations, and the ECCR cannot retrieve the registration changes during a restart or refresh operation. Instead, specify the LOCATION value for the PowerExchange Listener that manages the registrations and is connected to the PowerExchange Agent. Then, the updated registration information is available for ECCR processing.

LOCATION_CRG

The location of the registration file (CCT). The default is the LOCATION parameter value.

LOCATION_DM

The location of the DATAMAP file. The default is the LOCATION parameter value.

LOCATION_XDM

The location of the extraction maps. The default is the LOCATION parameter value.

NOTIFYCHANGES

For DB2 and Oracle sources, if you specify NOTIFYCHANGES=Y, any change to the schema for the table causes PowerExchange CDC to fail and log an error message.

For a DB2 for z/OS source, the DB2 ECCR ends abnormally after it reads the first change record for the table after the schema change.

For an Oracle source, Oracle CDC fails and logs an error message in the following situations:

- A change record for a registered table contains a column that is not registered for change capture.
- A change record does not contain any column that is registered for change capture.

For Oracle CDC, if a table definition changes in a way that is compatible with the PowerExchange capture registration, Oracle CDC continues to capture changes for that table. For example, if the length of a character column decreases but the capture registration does not reflect this change, Oracle CDC continues to capture changes for the table.

Conversely, if the data type of a column changes from numeric to character and the capture registration is not updated to reflect this change, Oracle CDC continues to capture changes for the table until it encounters the first change record that contains nonnumeric data for the column. Oracle CDC then fails and logs an error message.

For DB2 and Oracle sources, default is **Y**. For other data sources, default is **N**, which causes this parameter to be ignored.

OUTPUT

The location and file name of the report that DTLUCBRG generates. For example:

```
OUTPUT=c:\pwx\outfile.txt
```

If the path includes names with spaces, enclose the path in quotation marks.

On z/OS, the report is directed to the SYSPRINT DD output by default. On Windows, the report is directed to STDOUT by default.

PWD

A clear text password for the specified user ID. For access to an i5/OS or z/OS location, you can enter a valid PowerExchange passphrase instead of a password. An i5/OS passphrase can be from 9 to 31 characters in length. A z/OS passphrase can be from 9 to 128 characters in length. A passphrase can contain the following characters:

- Uppercase and lowercase letters
- The numbers 0 to 9
- Spaces
- The following special characters:

```
' - ; # \ , . / ! % & * ( ) _ + { } : @ | < > ?
```

Note: The first character is an apostrophe.

Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.

If a passphrase contains spaces, you must enclose it with double-quotation marks ("), for example, "This is a passphrase". If a passphrase contains special characters, you must enclose it with triple double-quotation characters ("""), for example, """This passphrase contains special characters ! % & * . """ . If a passphrase contains only alphanumeric characters without spaces, you can enter it without delimiters.

To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the *PowerExchange Reference Manual*.

Note: On z/OS, a valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.

Do not also specify EPWD.

REPLACE

Whether to replace existing inactive registrations. Options are:

- **Y.** Replace existing inactive registrations that match the mask specified in the TABLE parameter.
- **N.** Do not replace any existing registrations. Add registrations for tables that match the mask and do not yet exist.

Default is **N**.

REPLACEACTIVE

Whether to replace existing active registrations. Options are:

- **Y.** Replace existing active registrations that match the mask specified in the TABLE parameter. For this replacement to occur, the REPLACE parameter must also be set to Y.
- **N.** Do not replace active registrations.

Default is **N**.

REUSECRGNAME

Whether the DTLUCBRG retains the current names of existing capture registrations or renames them. Options are:

- **Y.** Retain the current names of existing registrations.
- **N.** Rename existing registrations by using the CRGPREFIX and sequential number format.

Default is **N**.

RPTCOLS

Whether to rename any existing capture registrations or retain their current names. Options are:

- **Y.** Retain the current names of existing registrations.
- **N.** Rename the existing registrations by using the CRGPREFIX and sequential number format.

Default is **Y**.

STATUS

The status to set for the capture registrations that the DTLUCBRG utility creates. Options are:

- **A.** Create registrations with the status of Active.
- **I.** Create registrations with the status of Inactive. Later, you will need to activate the registrations to make them eligible for change capture use.

For Microsoft SQL Server sources, if you specify UPDATESTATUS=Y in the MSSOPTS parameter, this STATUS parameter resets the status of all registrations that match the filter criteria you specify.

TABLE

A mask that restricts the source tables for which the DTLUCBRG utility will create capture registrations.

- For relational database tables, the mask can be specified in the following format:

owner_or_schema.tablename

Within this mask, the schema name and table name can each be up to 128 characters in length.

- For nonrelational sources, the mask includes the data map name and can be specified in the following format:

schema.mapname

Optionally, you can define a TABLE parameter value in NRDB or NRDB2 format that includes a table name to narrow the selection of tables when multiple tables are defined in a data map but not all of the tables need to be registered.

The following example shows the three-tier NRDB format, which uses period (.) separators between the schema name, map name, and table name:

SCHEMA.mapname.tablename

The following example shows the two-tier NRDB2 format, which uses an underscore (_) separator between the map name and table name:

SCHEMA.mapname_tablename

To represent one or more characters in any part of the parameter value, use the asterisk (*) wildcard. For example, the following mask matches all relational tables that have the owner "OWNAB" and a table name that starts with "T":

*OWNAB.T**

The following example shows a parameter value in NRDB2 format that contains a wildcard mask in both the map name and table name. The parameter value matches map names that start with an "M", and table names that start with "T".

*SCHEMA1.M*_T**

The following example shows the same masks in NRDB format:

SCHEMA1.M.T**

TESTRUN

Whether to perform a test run of the DTLUCBRG utility. Options are:

- **Y.** Perform a test run to determine the capture registrations to be updated or added. The test run does not actually update or add any registration.
- **N.** Do not perform a test run. When you run the utility, it will update or add capture registrations.

Default is **Y**.

UID

A user ID that allows access to the source. The requirement for this parameter depends on both the data source that is being registered and the value of the SECURITY statement in the PowerExchange DBMOVER configuration file.

For a source on a supported Linux, UNIX, or Windows system, if you have enabled PowerExchange LDAP user authentication and, if applicable, disabled relational pass-through authentication, the user ID is the enterprise user name. For more information, see the *PowerExchange Reference Manual*.

If you specify a user ID, also specify either a PWD or EPWD value.

File Format for the INPUT_FN Option

Sample file format for the CREATEBICI=(COLUMNS=FILE, INPUT_FN=*file_name*) parameter.

If you include the CREATEBICI parameter in the input to DTLUCBRG, and specify the COLUMNS option as FILE, you must include an input file that contains the columns for which before-image or change-indicator columns are generated in the extraction maps. The following example shows the format of the entries in the input file:

```
* Comment line. This is ignored by processing.  
MYSCHEMA.MY_REGISTERED_TABLE.TABLE_SELECTED_COLUMN_01  
MYSCHEMA.MY_REGISTERED_TABLE.TABLE_SELECTED_COLUMN_02  
MYSCHEMA.MY_REGISTERED_TABLE.TABLE_SELECTED_COLUMN_03
```

If the file specified by the INPUT_FN option is not present, the utility issues a warning message, and no before-image or change-indicator columns are generated.

Expression-Field Functions That Support CREATEBICI Processing

If you specify user-defined fields with expressions that are associated with PowerExchange functions in the CREATEBICI parameter, ensure that the fields are associated with functions that support before-image (BI) buffering. If the functions do not support BI buffering, the DTLUCBRG utility does not create the BI and CI columns.

The following table lists functions that support BI buffering for Adabas, IDMS, IMS, i5/OS sequential, and VSAM ESDS and RRDS data sources:

Function	Returns
GetDatabaseKey	Database key for a record or a segment.
GetDBKey	Alias for GetDatabaseKey.

The following table lists functions that support BI buffering for IDMS sources only:

Function	Returns
GetDbKeyOfFirstMember	Database key of the first member of the specified set.
GetDbKeyOfLastMember	Database key of the last member of the specified set.
GetDbKeyOfNextMember	Database key of the next member record of the specified set.
GetDbKeyOfOwner	Database key of the owner of an IDMS set name.
GetDbKeyOfPriorMember	Database key of the previous member record of a specified set.
GetFullDbKey	Fully qualified database key of an IDMS record, which includes the 4-byte concatenated page group and radix identifier prefix.
GetFullDbKeyOfFirstMember	Fully qualified database key of the first member of the specified set, which includes the 4-byte concatenated page group and radix identifier prefix.
GetFullDbKeyOfLastMember	Fully qualified database key of the last member of the specified set, which includes the 4-byte concatenated page group and radix identifier prefix.

Function	Returns
GetFullDbKeyOfNextMember	Fully qualified database key of the next member of the specified set, which includes the 4-byte concatenated page group and radix identifier prefix.
GetFullDbKeyOfOwner	Fully qualified database key of the owner of an IDMS owner record, which includes the 4-byte concatenated page group and radix identifier prefix.
GetFullDbKeyOfPriorMember	Fully qualified database key of the previous member of the specified set, which includes the 4-byte concatenated page group and radix identifier prefix.

For more information about user-defined functions, see Appendix A in the *PowerExchange Navigator User Guide*.

DTLUCBRG Utility Source-Specific Parameters

For Adabas, IMS, Microsoft SQL Server, MySQL, Oracle, PostgreSQL, and VSAM data sources, the DTLUCBRG utility requires additional information to update registrations. Provide this information in the source-specific parameters.

These parameters have names that begin with the DBTYPE value and end with a suffix of OPTS. Each one has one or more subparameters.

The following list describes the source-specific parameters and subparameters:

ADAOPTS

Specifies parameters for Adabas sources. To use these parameters, the DBTYPE parameter must be set to ADA in the input parameter file. Otherwise, the utility issues an error message.

Syntax:

```
ADAOPTS=(FileNo=file number,DBID=dbid)
```

Subparameters:

FileNo

Required. The Adabas file number. No default is provided.

DBID

Required. The Adabas DBID. No default is provided.

IMSOPTS

Specifies parameters for IMS sources. To use these parameters, the DBTYPE parameter must be set to IMS in the input parameter file. Otherwise, the utility issues an error message.

Syntax:

```
IMSOPTS=([TYPE={SYN|LOG}][,DBDNAME=DBC_name][,IMSID=subsystem_id][,PRIMDSN=dataset_name])
```

Subparameters:

TYPE

Optional. The type of PowerExchange IMS CDC processing that you use. Options are:

- **SYN** for IMS synchronous CDC.
- **LOG** for IMS log-based CDC.

Default is SYN.

DBDNAME

Optional. The database name from the DBD. Default is From Datamap.

IMSID

Optional. The IMS subsystem ID, which matches the value in the IMSID statement in the DBMOVER member of the RUNLIB library. No default is provided.

PRIMDSN

Optional. The primary data set name. No default is provided.

MSSOPTS

Specifies parameters for Microsoft SQL Server sources. To use these parameters, the DBTYPE parameter must be set to MSS in the input parameter file. Otherwise, the DTLUCBRG utility issues an error message.

Syntax:

```
MSSOPTS=(DBSERVER=database_server,DBNAME=database_name,[UPDATESTATUS={Y|N}])
```

Subparameters:

DBSERVER

Required. The name of the database server. You can optionally include a port number for the server in the following format:

```
DBSERVER="database_server,port_number"
```

In this case, the double-quotation marks are required.

No default is provided.

DBNAME

Required. The name of the database that contains the tables from which changes are captured. Default is From Datamap.

UPDATESTATUS

Optional. Indicates whether the utility can change the status of multiple registrations in one operation and delete and rebuild the associated publications. Use this parameter when you need to switch the status of many registrations at one time, or when you need to make DDL changes to many source tables and do not want to manually re-create the registrations and regenerate the publications.

If you specify UPDATESTATUS=Y, the utility performs the following actions, depending on the STATUS setting:

- If STATUS is set to I, the active registrations that you select are set to inactive and the associated publications are deleted.
- If STATUS is set to A, the inactive registrations that you select are set to active and the associated publications are automatically rebuilt based on the existing registrations.

You can filter the registrations for the utility to process by specifying the CONDTYPE, TABLE, and CRGNAME or CRGPREFIX parameters.

Default is N.

MYSOPTS

Specifies a parameter for MySQL sources. To use this parameter, the DBTYPE parameter must be set to MYS in the input parameter file. Otherwise, the utility issues an error message.

Syntax:

```
MYSOPTS=(DBSERVER="{database_server_name|localhost} [{,port_number|3306}])"
```

Subparameter:

DBSERVER

Required. Enter the server name or IP address of the MySQL server where the source database is located. If the MySQL server is on the local host where the DTLUCBRG utility runs, you can enter **localhost**. Optionally, you can append the port number if you use a port other than the default port of 3306, for example, DBSERVER="localhost,3456".

ORAOPTS

Specifies a parameter for Oracle sources. To use this parameter, the DBTYPE parameter must be set to ORA in the input parameter file. Otherwise, the utility issues an error message.

Syntax:

```
ORAOPTS=(DDLFILE=[path] [file_name])
```

Subparameter:

DDFILE

Required. The name of the file that stores ALTER DDL statements for supplemental log groups. You can specify the file name only or the full path and file name, for example, oraopts.sql or c:\sql\oraopts.sql. Alternatively, you can enter the subparameter with a blank value, DDFILE=, to use the default file name and directory.

If the parameter value includes spaces, do not use quotes to delimit the path and file name. If you specify only a path, the utility returns an error.

Default value is dtlucbrg_ora.sql.

PGSOPTS

Specifies parameters for PostgreSQL sources. To use these parameters, the DBTYPE must be set to PGS in the input parameter file. Otherwise, the DTLUCBRG utility issues an error message.

Syntax:

```
PGSOPTS=(DBSERVER=database_server,DBNAME=database_name)
```

Subparameters:

DBSERVER

Required. The name of the database server. You can optionally include a port number for the server in the following format:

```
DBSERVER="database_server,port_number"
```

In this case, the double-quotation marks are required.

No default is provided.

DBNAME

Required. The name of the database that contains the tables from which changes are captured.

VSMOPTS

Specifies a parameter for VSAM sources. To use this parameter, the DBTYPE parameter must be set to VSM in the input file. Otherwise, the utility issues an error message.

Syntax:

```
VSMOPTS=(FNAME=file_name)
```

Subparameters:

FNAME

Required. The name of the source VSAM data set.

RELATED TOPICS:

- [“DTLUCBRG Utility Parameters” on page 113](#)

Specifying Multiple Sets of Parameters on the DTLUCBRG Utility

Multiple sets of parameters can be placed in the same parameters file. These sets must be separated with a ‘;’ positioned on a new line between the sets of parameters. For example, on Linux, UNIX, or Windows, you could include the following lines:

```
DBTYPE DB2
TABLE DTL*
OUTPUT=c:\dtlucdb2.txt
etc. ...
;
DBTYPE DB2
TABLE PWX*
OUTPUT=c:\dtlucdb2_1.txt
etc. ...
```

Note: To see output from each set of parameters on Linux, UNIX, or Windows, define a different file for OUTPUT=. On z/OS, you cannot specify multiple output files. Each set of parameters is appended to the SYSPRINT DD output.

Sample Input for the DTLUCBRG Utility

The following sample input registers all DB2 tables on the DSN1 subsystem that have an owner name beginning with the characters "DTL":

```
DBTYPE DB2
TABLE DTL*
CONDTYPE NONE
INSTANCE DSN1
LOCATION MP3000
LOCATION_CRG MP3000
LOCATION_DM MP3000
LOCATION_XDM MP3000
CRGPREFIX DB2
TESTRUN N
STATUS A
UID dtlusr
PWD dtlusr
OUTPUT=c:\dtlucdb2.txt
REPLACE Y
REPLACEACTIVE Y
RPTCOLS N
```

The TABLE parameter works with the REPLACE and REPLACEACTIVE parameters to indicate that any active or inactive registrations that match the TABLE mask will be replaced.

The CONDTYPE setting of NONE indicates that data will not be available for PowerExchange Condense or PowerExchange Logger for Linux, UNIX, and Windows processing. For information about PowerExchange Condense or the PowerExchange Logger, see the PowerExchange CDC guide for your operating system.

The INSTANCE parameter specifies the DB2 subsystem name.

The LOCATION and LOCATION_xxx parameters indicate that the MP3000 system will contain the target data and the registration, data map, and extraction map files. Verify that each LOCATION parameter value maps to a NODE statement in the DBMOVER configuration file.

The STATUS and CRGPREFIX parameters indicate that the registrations will be created with a status of Active and the prefix "DB2."

The RPTCOLS parameter setting of NO indicates that the DTLUCBRG report output will show table names only, without column information. The OUTPUT parameter indicates that the report output will be written to the dtlucdb2.txt file.

RELATED TOPICS:

- [“DTLUCBRG Utility with RPTCOLS=N Report Description” on page 130](#)
- [“DTLUCBRG Utility with RPTCOLS=Y Report Description” on page 131](#)

Sample Input Creating Before-Image and Change-Indicator Columns

This sample input shows how to configure the DTLUCBRG utility parameter to create an extraction map that includes before-image and change-indicator columns for source data columns.

Input Parameters to Create BI and CI Columns

The following sample input parameters include the CREATEBICI parameter for creating before-image and change-indicator columns:

```
DBTYPE DB2
TABLE DTL*
CONDTYPE NONE
CREATEBICI=(COLUMNS=PKFK, MAXCOLS=15000, EXTINFO=BICI)
INSTANCE DSN1
LOCATION MP3000
LOCATION_CRG MP3000
LOCATION_DM MP3000
LOCATION_XDM MP3000
CRGPREFIX DB2
TESTRUN N
STATUS A
UID dtlusr
PWD dtlusr
OUTPUT=c:\dtlucdb2.txt
REPLACE Y
REPLACEACTIVE Y
RPTCOLS Y
```

The TABLE parameter works with the REPLACE and REPLACEACTIVE parameters to cause any active or inactive registrations that match the TABLE mask to be replaced.

The CREATEBICI parameter is configured to generate both before-image and change-indicator columns for up to 15000 primary key and foreign key columns in each table that matches the TABLE mask.

The RPTCOLS parameter setting of YES causes the DTLUCBRG report output to show column information. The BI and CI columns created in the extraction map are included in the report.

The OUTPUT parameter causes the report output to be written to the dtlucdb2.txt file.

DTLUCBRG Code Page Processing

DTLUCBRG disregards the CODEPAGE statement in the DBMOVER configuration file and instead uses the code page that is used to store registrations and capture data map metadata, as follows:

- IBM037 on i5/OS
- IBM1047 on z/OS
- UTF-8 on Linux, UNIX, or Windows

Whenever the DBMOVER CODEPAGE statement is overridden, PowerExchange issues the following message:

```
Changed client code pages to name (internal_code_page_number)
```

On i5/OS and z/OS, PowerExchange supports table and column names with characters that are present in code pages IBM037 and IBM1047, respectively. Accented characters that are not in the code pages are not supported. On Linux, UNIX, and Windows, all characters are supported.

When the PowerExchange Listener retrieves data, code page conversion of SQL and data is performed automatically. For example, a PowerExchange Listener on z/OS might need to use a particular SQL code page to meet the requirements of a DB2 subsystem. SQL is converted to the required code page and is received by the Listener ready for use by DB2. Column data is described according to the DB2 CCSIDs and sent back to DTLUCBRG, which converts it to the required metadata code page.

Code Page Processing for DB2 Registrations in Local Mode on z/OS

Because a PowerExchange Listener is not involved, code page processing for DB2 registrations in local mode on z/OS does not trigger automatic SQL code page conversion. In this case, DTLUCBRG performs the required code page conversion and issues the following message:

```
Using codepage code_page_name (code_page_number for table names in DB2 subsystem subsystem.
```

Note that because DTLUCBRG uses connection pooling when using a PowerExchange Listener, performance is only slightly slower than when running locally.

Running the DTLUCBRG Utility

It is strongly advised that the utility is run with TESTRUN=Y initially to assess the scope of the changes and additions to registration resulting from a particular run. After you see the changes reported by the TESTRUN=Y execution, change TESTRUN to N and run to see the changes take effect.

Running the DTLUCBRG Utility on i5/OS

On i5/OS, run the utility by entering the following command:

```
call dtlucbrg parm('cs=filepath1/filepath2(myparmfile)')
```

Where *myparmfile* contains the DTLUCBRG control statements.

Running the DTLUCBRG Utility on Linux, UNIX, and Windows

The input parameters are supplied in the dtlucbrg.txt file by default. If the parameters are in this file, run the utility by entering DTLUCBRG on the command line. Parameters can be supplied in a file of another name. The report is written to the location specified in the OUTPUT parameter.

Note: If PowerExchange accesses Microsoft SQL Server on a Linux operating system, the LD_LIBRARY_PATH environment variable on the Linux system must specify the value \$PWX_HOME/ODBC7.1/lib for the DTLUCBRG utility to run. For more information about setting the LD_LIBRARY_PATH environment variable, see the *PowerExchange Installation and Upgrade Guide*.

DTLUCBRG Utility on Linux and UNIX Syntax

On Linux and UNIX, run the utility by entering DTLUCBRG on the command line as follows:

```
dtlucbrg CS=/MyParms/PWX/ucbrgtest.txt
```

DTLUCBRG Utility on Windows Syntax

On Windows, to run with a specified file path and name, use the following syntax:

```
c:\>dtlucbrg CS=C:\MyParms\PWX\ucbrgtest.txt
```

If the path or file name contains embedded blanks, use the following syntax:

```
c:\>dtlucbrg CS="C:\MyParms\PWX\In Quotes for Embedded Blanks.txt"
```

Running the DTLUCBRG Utility on z/OS

The following JCL provides example statements to use when you run this utility on z/OS:

```
//DTLUSRRG JOB 'DTLSETFL',MSGCLASS=X,NOTIFY=&SYSUID,
//          CLASS=A,REGION=64M
//*****
//*
//* RUN BATCH REGISTRATION UTILITY
//*
//*****
//INCS1 INCLUDE MEMBER=GENBULK
//***
//RUN      EXEC PGM=DTLUCBRG
//*
//*
//STEPLIB DD DISP=SHR,DSN=&HLQ..LOADLIB
//          DD DISP=SHR,DSN=&HLQ..LOAD
//          DD DISP=SHR,DSN=&SCERUN
//          DD DISP=SHR,DSN=&DB2LOAD
//          DD DISP=SHR,DSN=&DB2EXIT
//*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//***
//SYSIN DD *
DBTYPE DB2
TABLE DTLUSR.DTL*
CONDTYPE NONE
INSTANCE DSN1
LOCATION node1
LOCATION_CRG node1
LOCATION_DM node1
LOCATION_XDM node1
CRGPREFIX DB2
TESTRUN N
STATUS A
UID <logonid>
PWD xxxxxx
```



```

REPLACE Y
REPLACEACTIVE Y
RPTCOLS N
/*
/* - other parms
/* EPWD
/* REUSECRGNAME
/*
/* CDC Datasets - need to be open if CDC to be used
/*
//DTLMSG DD DISP=SHR,DSN=&HLQ..DTLMSG
//DTLCFG DD DISP=SHR,DSN=&RUNLIB (DBMOVER)
//DTLKEY DD DISP=SHR,DSN=&RUNLIB (LICENSE)
//DTLSGN DD DISP=SHR,DSN=&RUNLIB (SIGNON)
//DTLLOG DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//CEEDUMP DD SYSOUT=*

```

The following JCL provides example statements to use when you run this utility on z/OS with the CREATEBICI parameter enabled:

```

//DTLUSRRG JOB 'DTLSBC1',MSGCLASS=X,NOTIFY=&SYSUID,
// CLASS=A,REGION=64M
//*****
/*
/* RUN BATCH REGISTRATION UTILITY
/*
/******
//INCS1 INCLUDE MEMBER=GENBULK
//***
//RUN EXEC PGM=DTLUCBRG
/*
/*
//STEPLIB DD DISP=SHR,DSN=&HLQ..LOADLIB
// DD DISP=SHR,DSN=&HLQ..LOAD
// DD DISP=SHR,DSN=&SCERUN
// DD DISP=SHR,DSN=&DB2LOAD
// DD DISP=SHR,DSN=&DB2EXIT
/*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//***
//SYSIN DD *
DBTYPE DB2
TABLE DTLUSR.DTL*
CONDTYPE NONE
INSTANCE DSN1
LOCATION node1
LOCATION_CRG node1
LOCATION_DM node1
LOCATION_XDM node1
CRGPREFIX DB2
TESTRUN N
STATUS A
UID <logonid>
PWD xxxxxx
CREATEBICI=(COLUMNS=ALL, MAXCOLS=15000, EXTINFO=BICI, INPUT_FN=Y, OUTPUT_FN=Y, ERROR_FN=Y)
REPLACE Y
REPLACEACTIVE Y
RPTCOLS Y
/*
/* - other parms
/* EPWD
/* REUSECRGNAME
/*
/* CDC Datasets - need to be open if CDC to be used
/*
//DTLMSG DD DISP=SHR,DSN=&HLQ..DTLMSG
//DTLCFG DD DISP=SHR,DSN=&RUNLIB (DBMOVER)
//DTLKEY DD DISP=SHR,DSN=&RUNLIB (LICENSE)
//DTLSGN DD DISP=SHR,DSN=&RUNLIB (SIGNON)
//DTLLOG DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSOUT DD SYSOUT=*

```

```
//SYSPRINT DD SYSOUT=*
//CEEDUMP DD SYSOUT=*
//*
//*CREATEBICI input, output, and error files
//*
//*Note that these data sets must be preallocated as PDSE,
//*DSNTYPE(LIBRARY), with a record length of 255
//*
//UCBRGFIP DD DISP=SHR,DSN=&RUNLIB(UCBRGINP)
//UCBRGFOP DD DISP=SHR,DSN=&RUNLIB(UCBRGOUT)
//UCBRGFER DD DISP=SHR,DSN=&RUNLIB(UCBRGERR)
```

DTLUCBRG Utility Example Reports

You can specify the output of the DTLUCBRG utility by setting the RPTCOLS parameter.

The following topics show example reports with the RPTCOLS parameter set to **Y** and **N**.

DTLUCBRG Utility with RPTCOLS=N Report Description

The following example report provides information at the table level. It does not include column detail because RPTCOLS=N. The header contains the TESTRUN setting, which should be checked to ensure that the utility ran in the expected mode. After the header, the report shows the contents of the input parameter file.

```
2005-01-24 11:07:16 DTLUCBRG REGISTRATION REPORT (TESTRUN=N)
CONDTYPE      = <None>
CRGPPREFIX    = <DB2>
DBTYPE        = <DB2>
INSTANCE      = <DSN1>
LOCATION        = <MP3000>
LOCATION_CRG    = <MP3000>
LOCATION_DM     = <MP3000>
LOCATION_XDM    = <DB3000>
OUTPUT        = <c:\dtlucdb2.txt>
REPLACE       = <Y>
REPLACEACTIVE = <Y>
REUSECRGNAME  = <N>
RPTCOLS       = <N>
STATUS        = <A>
TABLE         = <DTL*>
IMSOPTS: Not relevant for this run
ORAOPTS: Not relevant for this run
MSSOPTS: Not relevant for this run
ADAOPTS: Not relevant for this run
.
.
```

```
Processed Registration "db20030" Table
"db2.db2_ALLDT"
```

```
Old Registration Name "db20030". Old condense type "None". Old status "A". Old version
"1".
```

RegName	Old RegName	Table-name	Old Cond	Old Status	Vers >1
db20008	db2captc	DTLUSR.DTLRESTART	Part	A	-
db20009	db20001	DTLUSR.DTLSTATUS	None	A	-
db20010	db20002	DTLUSR.DTLTST4	None	A	-
db20011	db20003	DTLUSR.DTLTST5	None	A	-
db20012	db20004	DTLUSR.DTLTST6	None	A	-
db20013	db20005	DTLUSR.DTLTST8	None	A	-
db20014	db20006	DTLUSR.DTLTST9	None	A	-

```

Summary of registrations created with status ACTIVE and
condense type NONE
No of registrations created = 0
No of registrations updated = 7
No of existing registrations not matching update parameters: = 0
2005-01-24 11:08:13 END OF DTLUCBRG REGISTRATION REPORT

```

The following table describes the fields in the example summary of registrations report:

Report Field	Description
RegName	The name of the new registration.
Old RegName	This value appears if the old registration name was replaced because REUSECRGNAME=N was specified.
Table-name	The table that is being registered for change data capture.
Old Cond	The Condense option in the old registration. This value appears only if the old registration was replaced.
Old Status	The Status value in the old registration. This value appears only if the old registration was replaced.

DTLUCBRG Utility with RPTCOLS=Y Report Description

The following example report shows the additional information that is generated when you run the DTLUCBRG utility with the RPTCOLS parameter set to Y:

```

-----
db20030 db20023 DTLUSR.DTLSTATUS          None A -
-Column Name -----Type-----Precision--Scale--Nulls-Key-----
TABLE_NAME          VARCHAR          255    0    N    Y
STATUS              CHAR              20     0    N    N
STATUS_REASON       CHAR              20     0    N    N
APPLY_SEQUENCE      VARCHAR          255    0    Y    N
RESTART_POINT       VARCHAR          255    0    Y    N
-----

```

If you use the CREATEBICI parameter to generate before-image or change-indicator columns, the GEN_BI and GEN_CI columns are added to the report to indicate whether a data column has an associated before-image or change-indicator column.

```

2018-06-21 15:31:05 DTLUCBRG REGISTRATION REPORT
(TESTRUN=N)
.
CONDTYPE           = <None>
CRGNAME            = <kafkaall>
DBTYPE             = <VSM>
FASTLOAD           = <N>
INSTANCE           = <INST6>
LOCATION            = <node1>
LOCATION_CRG        = <node1>
LOCATION_DM         = <node1>
LOCATION_XDM        = <node1>
NOTIFYCHANGES     = <N>
OUTPUT             = <>
REPLACE            = <Y>
REPLACEACTIVE      = <Y>
REUSECRGNAME       = <Y>
RPTCOLS            = <Y>
STATUS             = <A>
TABLE              = <kafsvr.vsamall.ALIDataTypes>
.
IMSOPTS: Not relevant for this run
ORAOPTS: Not relevant for this run
MYSOPTS: Not relevant for this run
MSSOPTS: Not relevant for this run
ADAOPTS: Not relevant for this run

```

```

CREATEBICI:    COLUMNS = <ALL>
              MAXCOLS = <500>
              EXTINFO = <BICI>
              INPUT_FN = <N>
              OUTPUT_FN = <N>
              ERROR_FN = <N>
.
.
Processed Registration "kafkaall" Table "kafsvr.vsamall_ALLDataTypes"
Old Registration Name "kafkaall". Old condense type "None". Old status "A". Old version "1".
.
Column Name ----- Type ----- -- Precision  Scale Nulls Key BI CI
BIN9                BIN          9          0 N      N Y Y
CHAR4                CHAR          4          0 N      N Y Y
Double              DOUBLE         0          0 N      N Y Y
Float                DOUBLE         0          0 N      N Y Y
Num16_0              NUM16          0          0 N      N Y Y
Num16_6              NUM16          0          5 N      N Y Y
NUM16U_0             NUM16U         0          0 N      N Y Y
NUM16U_5             NUM16U         0          5 N      N Y Y
NUM32_0              NUM32          0          0 N      N Y Y
NUM32_10             NUM32          0         10 N      N Y Y
NUMCHAR1             NUM8           0          0 N      N Y Y
NUMCHAR100           NUMCHAR        100         0 N      N Y Y
NUMCHAR100_99        NUMCHAR        100         0 N      N Y Y
PACKED31_0           PACKED         31          0 N      N Y Y
PACKED31_31          PACKED         31         31 N      N Y Y
PACKED5_3            PACKED          5          3 N      N Y Y
SPACKED31_0          UPACKED        31          0 N      N Y Y
SPACKED31_31         UPACKED        31         31 N      N Y Y
SPACKED_3            UPACKED          5          3 N      N Y Y
UPACKED31_0          UPACKED        31          0 N      N Y Y
UPACKED              UPACKED        31         31 N      N Y Y
UPACKED5_3           UPACKED          5          3 N      N Y Y
UZONED31_0           UZONED         31          0 N      N Y Y
UZONED31_31          UZONED         31         31 N      N Y Y
VARBIN44             VARBIN         44          0 Y      N Y Y
VARCHAR1             VARCHAR          1          0 Y      N Y Y
VARCHAR255           VARCHAR        255          0 Y      N Y Y
VARCHAR4096          VARCHAR       4096          0 Y      N Y Y
ZONED31_0            ZONED          31          0 N      N Y Y
ZONED31_31           ZONED          31         31 N      N Y Y
ZONED5_3             ZONED           5          3 N      N Y Y
-----
.
=====
Summary of registrations created with status ACTIVE and condense type NONE
No of registrations created = 0
No of registrations updated = 1
No of existing registrations not matching update parameters = 0
.
2018-06-21 15:31:09 END OF DTLUCBRG REGISTRATION REPORT
=====

```

The following table describes the fields for the extended report format:

Field	Description
Column Name	Column name
Type	Data type
Precision	Length of the column
Scale	Decimal places
Nulls	Indicates whether the column can contain a null. Valid values: Y or N.
Key	Indicates whether the column is a key column. Valid values: Y or N.

Field	Description
GEN_BI	Indicates whether a before-image column is generated for the column. Valid values: Y or N.
GEN_CI	Indicates whether a change-indicator column is generated for the column. Valid values: Y or N.

Column information is displayed immediately after the relevant table registration information.

Notes:

- Table and column names might be truncated in the report.
- GEN_BI and GEN_CI headings are shown only if the CREATEBICI parameter is specified.

DTLUCBRG Utility Usage Notes

Review the following considerations:

- When you run the DTLUCBRG utility, you might receive error messages that imply the registrations that are being created might not be usable. For example:

```
PWX-09702  Oracle ID xxxx not found in configuration
```

To avoid the creation of unusable registrations, perform the following actions:

- Correct the error that the message reports and then re-run the utility. Repeat this process until the utility generates no error messages. It is particularly important to eliminate errors that result from the lookup of existing registrations, because these errors might result in unusable registrations even if no entries are found.
- First attempt to create a small volume of registrations.
- Include the following parameter in the utility input to perform a test run of the utility without creating the registrations:

```
TESTRUN Y
```

After you perform a test run without errors for a small volume of registrations, try the test run again for the complete set of registrations that you want to create. After this run completes without errors, specify TESTRUN N in the utility input and run the utility again to create the registrations. Default is TESTRUN Y.

- For z/OS, specify a minimum REGION size of 64M to accommodate memory requirements for utility processing.
- If you use the DTLUCBRG utility to create capture registrations for data sources on z/OS but the utility does not find the required source data maps in the DATAMAPS member on the LOCATION_DM node, the utility does not create the registrations and does not report the error. Instead, the utility appears to end successfully with RC=0. To avoid this problem, ensure that the data maps for the source objects exist in the DATAMAPS member before you run the utility.
- In some cases, the DTLUCBRG utility might not generate the BI and CI columns even though the CREATEBICI parameter is specified. For example, the utility does not generate BI and CI columns for DB2 for z/OS LOB columns. In this case, the utility ignores the attempt to create the columns in the extraction map and writes a warning message to the output file.
- If you use the DTLUCBRG utility with DB2 for i5/OS sources, you can specify either a flat physical file or source physical file in the INPUT_FN, OUTPUT_FN, or ERROR_FN keyword of the CREATEBICI parameter.

These keywords point to input, output, and error files. If you want to specify a member within a file, enclose the entire file and member name specification in double quotation marks. For example:

```
CREATEBICI=(COLUMNS=ALL, EXTINFO=BICI, INPUT_FN=MYLIB/MYFILEIN,  
OUTPUT_FN=MYLIB/MYFILEOUT, ERROR_FN="MYLIB/MYSRCFILE(ERROR_MBR) ")
```

- If you are generating before-image or change-indicator columns and the COLUMNS keyword identifies user-defined columns with expressions that invoke PowerExchange functions, ensure that the functions support BI buffering. If a function does not support BI buffering, PowerExchange does not generate BI or CI columns for the field. For more information, see [“Expression-Field Functions That Support CREATEBICI Processing ” on page 121](#).
- For z/OS and IBMi operating systems, the extraction maps generated by the utility might exceed the default record lengths used for the system. For z/OS the default record length is 88 characters and for IBMi it is 184 characters. If you encounter errors when the system attempts to write the output from the utility, you might need to modify the utility input file or JCL SYSIN DD statement to override the default record length.
- If PowerExchange accesses Microsoft SQL Server on a Linux operating system, the LD_LIBRARY_PATH environment variable on the Linux system must specify the value \$PWX_HOME/ODBC7.1/lib for the DTLUCBRG utility to run. For more information about setting the environment variable, see the *PowerExchange Installation and Upgrade Guide*.
- The DTLUCBRG utility cannot create capture registrations for Oracle global temporary tables. PowerExchange cannot capture change data from these tables because the table data is not available in the Oracle redo logs. The data is stored in temporary tablespaces for private use and deleted when the database session ends. If you try to register a global temporary table with the DTLUCBRG utility, the utility produces no registration for the table.

CHAPTER 9

DTLUCDEP - CDEP Maintenance Utility

This chapter includes the following topics:

- [DTLUCDEP Utility Overview, 135](#)
- [Supported Operating Systems for the DTLUCDEP Utility , 136](#)
- [Control Statement Syntax for the DTLUCDEP Utility, 136](#)
- [Control Statement Parameters for the DTLUCDEP Utility, 136](#)
- [Running the DTLUCDEP Utility on i5/OS, 138](#)
- [Running the DTLUCDEP Utility on Linux, UNIX, and Windows, 138](#)
- [Running the DTLUCDEP Utility on z/OS, 138](#)
- [DTLUCDEP Utility on i5/OS Example, 140](#)
- [DTLUCDEP Utility on Linux, UNIX, and Windows Example, 140](#)
- [DTLUCDEP Utility on z/OS Example, 141](#)

DTLUCDEP Utility Overview

When you run PowerExchange change capture processes, you might need to delete obsolete or unnecessary applications and extractions from your PowerExchange Capture Extraction Process Control (CDEP) file.

Use the DTLUCDEP utility to modify or print the contents of the CDEP file. This file contains information about the change capture extraction processes that have run, timings, and input. The CDEP file is written to or read by the extraction process to establish the starting point for an extraction.

Warning: It is extremely important that this utility is used appropriately as any modifications performed on the CDEP file are irreversible. This could mean that starting points for your change capture processes may be lost.

It is suggested that a backup copy of the CDEP file is taken before running the DTLUCDEP utility.

Supported Operating Systems for the DTLUCDEP Utility

The DTLUCDEP utility can run on the following operating systems:

- i5/OS
- UNIX and Linux
- Windows
- z/OS

Control Statement Syntax for the DTLUCDEP Utility

Use the following syntax for the DTLUCDEP utility control statements:

```
[USER user_ID {pwd password|EPWD epassword}]  
{PRINT|MODIFY} APPL {appname|ALL} days
```

Control Statement Parameters for the DTLUCDEP Utility

Use the DTLUCDEP definition file to control whether the DTLUCDEP utility prints or modifies CDEP information for an application. You can filter the resulting utility output based on a number of days.

The utility has the following parameters:

USER *user_ID*

If security checking is enabled, an operating system user ID.

For an application on a supported Linux, UNIX, or Windows system, if you have enabled PowerExchange LDAP user authentication, the user ID is the enterprise user name. For more information, see the *PowerExchange Reference Manual*.

{PWD *password*|EPWD *encrypted_password*}

A password or encrypted password for the specified user.

- **PWD.** A password for the specified user.
For access to i5/OS or z/OS, you can enter a valid PowerExchange passphrase instead of a password. An i5/OS passphrase can be from 9 to 31 characters in length. A z/OS passphrase can be from 9 to 128 characters in length. A passphrase can contain the following characters:
 - Uppercase and lowercase letters
 - The numbers 0 to 9
 - Spaces
 - The following special characters:

' - ; # \ , . / ! % & * () _ + { } : @ | < > ?

Note: The first character is an apostrophe.

Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.

If a passphrase contains spaces, you must enclose it with double-quotation marks ("), for example, "This is a passphrase". If a passphrase contains special characters, you must enclose it with triple double-quotation characters ("""), for example, """This passphrase contains special characters ! % & *. """". If a passphrase contains only alphanumeric characters without spaces, you can enter it without delimiters.

To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the *PowerExchange Reference Manual*.

Note: On z/OS, a valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.

- **EPWD.** An encrypted password for the specified user.

For access to i5/OS or z/OS, you can enter an encrypted PowerExchange passphrase instead of an encrypted password. Do not encrypt a passphrase that contains invalid characters, such as double-quotation marks, single quotation marks, or currency symbols.

{PRINT|MODIFY}

Specify one of the following keywords:

- **PRINT.** Prints the CDEP details for the specified application.
- **MODIFY.** Removes details for the specified application from the CDEP file based on the days parameter.

APPL

Set to APPL.

appname

Name of the application that you want to print or modify. To specify all applications, enter "ALL." To specify multiple applications with the same name pattern, include the asterisk (*) wildcard character, for example, LULU*.

days

The number of days of information that the command processes.

For example, the following statement removes all progress details for the application LULU01 that are more than 21 days old:

```
modify appl LULU01 21
```

The following statement prints all progress details for the application LULU01 for the previous 21 days:

```
print appl LULU01 21
```

To remove all details for a particular application use 0 force. For example:

```
modify appl LULU01 0 force
```

If the days parameter is not specified, the utility prints progress details for the last seven days or removes (modifies) details that are older than 40 days.

CDEP Definition Examples

The following are examples of CDEP definitions and meanings:

The following statement prints the progress details of all applications in the CDEP file that occurred within the previous 256 days:

```
print appl ALL 256
```

The following statement removes all progress details for the application LULU03 prior to the last 14 days:

```
modify appl LULU03 14
```

The following statement removes all details of the application LULU06:

```
modify appl LULU06 0 force
```

Running the DTLUCDEP Utility on i5/OS

To run the DTLUCDEP utility on i5/OS:

1. Verify the definitions in the CFG(DTLUCDEP) definition file.
2. Enter the following command:

```
SBMJOB CMD(CALL PGM(DTLLIB/DTLUCDEP) PARM('CS=DATALIB/CFG(DTLUCDEP)')) JOB(MYJOB)
JOB(DATALIB/DTLLIST) PRTDEV(*JOB) OUTQ(*JOB) CURLIB(*CRTDFT) INLLIB(*JOB)
```

Running the DTLUCDEP Utility on Linux, UNIX, and Windows

To run the DTLUCDEP utility on Linux, UNIX, and Windows:

1. Verify the definitions in the dtlucdep.txt definition file.
2. Enter the following command:

```
DTLUCDEP
```

Running the DTLUCDEP Utility on z/OS

PowerExchange provides sample JCL for the DTLUCDEP utility in the DTLUCDEP member of the RUNLIB library.

To run the DTLUCDEP utility on z/OS:

1. The following JCL statements are required to run the utility. Specify the DTLUCDEP definitions in-stream, as follows, or in a referenced PDS by using the DD statement.

```
//jobname JOB
//STEP1 EXEC PGM=DTLUCDEP
//*
/* or EXEC PGM=DTLUCDEP,PARM=('CS=DD:DTLUCDEP')
/* which uses the specified DD instead of sysin
/*
//STEPLIB DD DISP=SHR,DSN=&HLQ..LOADLIB
// DD DISP=SHR,DSN=&HLQ..LOAD
// DD DISP=SHR,DSN=&SCERUN
```

```

//DTLCACDE DD DSN=&HLQVS..CDEP,
//          DISP= (SHR)
//DTLMSG   DD DSN=&HLQ..DTLMSG,
//          DISP= (SHR)
//DTLCFG   DD DSN=&RUNLIB (DBMOVER) ,
//          DISP= (SHR)
//DTLKEY   DD DSN=&RUNLIB (LICENSE) ,
//          DISP= (SHR)
//DTLSGN   DD DSN=&RUNLIB (SIGNON) ,
//          DISP= (SHR)
//DTLLOG   DD SYSOUT=*
//DTLLOG01 DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//
//SYSIN DD *
        user DTLUSR  epwd A3164A3622798FDC
        print appl testapp
/*

```

The JCL statements are:

JOB

Initiates the job.

EXEC PGM=DTLUCDEP

Invokes the utility.

STEPLIB DD

Defines the PowerExchange LOAD library that contains the utility.

DTLCACDE DD

Defines the CDEP file.

DTLMSG DD

Defines the PowerExchange message file.

DTLCFG DD

Defines the DBMOVER configuration file.

DTLKEY DD

Defines the license key file.

DTLSGN DD

Defines the selective sign-on file.

DTLLOG DD

Defines the PowerExchange message log file. PowerExchange writes messages to this log file until the alternative logging subtask is initialized.

DTLLOG01 DD

If you enable alternative logging, defines the PowerExchange alternative message log file.

SYSOUT DD

Defines the destination of printed output.

SYSPRINT DD

Defines the print location for the report.

2. Verify the definitions in the JCL.
3. Submit the DTLUCDEP job.

DTLUCDEP Utility on i5/OS Example

The following output is an example of the results of the DTLUCDEP utility:

```
03/11/04 10:01:22                                POWEREXCHANGE/CFG (DTLUCDEP) CARDS
=====
user XXXXXX  pswd 889B042B53F132B7
  print appl ALL 60
Print of requested All Applications
      since 03/09/05 10:01:22
=====
Print of testdota : All Applications
-----
Application name=<testdota> AS4 Rsttkn=<1>
      Ainseq=<0>
First run started  =<03/06/13 16:26:19> ended <03/06/13 17:06:08>
      sequence      =<2A102FE20A3600000000000000000770
                    66F22A102FE20A3600000000000000000
                    077066F1>
      restart        =<D9D6C4E3C5E2E3F32A102FE20A360000
                    000000000000077066F0>
Last run started   =<03/06/13 16:26:19> ended <03/06/13 17:06:08>
      sequence      =<2A102FE20A36000000000000000000770
                    66F22A102FE20A3600000000000000000
                    077066F1>
      restart        =<D9D6C4E3C5E2E3F32A102FE20A360000
                    000000000000077066F0>
Current run started =<> ended <>
      sequence      =<000000000000000000000000000000000
                    00000000000000000000000000000000
                    00000000>
      restart        =<000000000000000000000000000000000
                    0000000000000000>
Tokens supplied by the token utility
  Registration name=<dot1.1> tag=<AS4RODTEST3dot11>
      sequence      =<2A2F96A18FC00000000000000000000000
                    00F02A2F96A18FC000000000000000000
                    000000F0>
      restart        =<D9D6C4E3C5E2E3F32A2F96A18FC00000
                    000000000000000000F0>
Print of progress for testdota since 03/09/05 10:01:22
  No progress for Application name=<testdota>
Print of testdotal : All Applications
-----
```

DTLUCDEP Utility on Linux, UNIX, and Windows Example

The output can be piped to a text file if required using the normal command line pipe option. For example:

```
DTLUCDEP > output.txt
```

The following output is an example of the results of the DTLUCDEP utility:

```
2.2.4      DTLUCDEP Example output from the utility
03/10/31 15:46:12                                V:\bin\dtlucdep.txt CARDS
=====
  print appl LULU03

Print of requested Application LULU03 only
      since 03/10/24 15:46:12
=====
Print of LULU03 : Application LULU03 only
```

```

=====
Application name=<LULU03>  Rsttkn=<0>
                        Ainseq=<0>
First run started  =<03/10/24 11:17:37> ended <03/10/24 11:18:04>
sequence          =<0000000002B9960000000002B995>
restart           =<0000000002B9944D5045584C5F535953
                    54454D5F564F4C554D455F534554>
Last run started  =<03/10/24 11:17:37> ended <03/10/24 11:18:04>
sequence          =<0000000002B9960000000002B995>
restart           =<0000000002B9944D5045584C5F535953
                    54454D5F564F4C554D455F534554>
Current run started =<> ended <>
sequence          =<0000000000000000000000000000>
restart           =<0000000000000000000000000000
                    0000000000000000000000000000>

Print of progress for LULU03 since 03/10/24 15:46:12
No progress for Application name=<LULU03>

```

DTLUCDEP Utility on z/OS Example

The following output is an example of the results of the DTLUCDEP utility:

```

03/11/04 12:04:51          SYSIN CARDS
=====
user DTLUSR  epwd A3164A3622798FDC
print appl testapp
modify appl all 40
Print of requested Application testapp only
since 03/10/28 12:04:51
=====
DTL-04558 Application Index data for <testapp> not found.
Application name=<testapp> does not exist
Modify for requested All Applications
before 03/09/25 12:04:51
=====

Modify of TESTRUN : All Applications

Modify of progress for TESTRUN
before 03/09/25 12:04:51
No progress for Application name=<TESTRUN>
MOD Application name=<TESTRUN>  Rsttkn=<0>  Ainseq=<0>

First run started  =<03/11/04 12:01:10> ended <03/11/04 12:01:45>
sequence          =<000000004F0200000000000000004D1B
                    00000000>
restart           =<C4D6C3D34040000000003D4700000000>
Last run started  =<03/11/04 12:02:46> ended <03/11/04 12:03:12>
sequence          =<000000004F0200000000000000004D1B
                    00000000>
restart           =<C4D6C3D34040000000003D4700000000>
Current run started =<> ended <>
sequence          =<0000000000000000000000000000
                    00000000>
restart           =<0000000000000000000000000000>
Application TESTRUN - 0 progress entries expired
Application name=<>
0 applications 0 progress entries expired

***** BOTTOM OF DATA *****

```

CHAPTER 10

DTLUCSR2 - IDMS SR2 and SR3 Records Utility

This chapter includes the following topics:

- [DTLUCSR2 Utility Overview, 142](#)
- [Running the DTLUCSR2 Utility, 143](#)

DTLUCSR2 Utility Overview

When an IDMS record no longer fits on its home page, IDMS relocates the record and generates a control record on the home page that points to the relocated record. The relocated record is known as the SR3 record, and the control record is known as the SR2 record. If an update or delete occurs on a SR3 record, the IDMS log-based ECCR needs to get the original record ID from the SR2 record to determine if the change is eligible for change capture. To enable the ECCR to find this information, run the DTLUCSR2 utility. The utility records the pairs of matching SR2 and SR3 records in an internal table. The ECCR can then perform a lookup on the table with the SR3 database key to find the matching SR2 record that contains the original record ID.

Run the DTLUCSR2 utility before you start the ECCR for the first time and after events that tend to generate SR2 and SR3 records. For example, run the utility after the following events:

- An IDMS REORG operation
- An IDMS dictionary migration utility (RHDCMIG1 and RHDCMIG2) run
- An alter table operation that adds one or more columns, or any other schema change that can increase the record size
- The following PowerExchange program logic errors, which are issued for an after image (AFTR) or before image (BFOR):

```
PWX-00999 Program logic error. Prog="program". Line=line_number. P1="UOW - SR3 AFTR  
hex_SR3_database_key, not found in hash table". P2=1
```

```
PWX-00999 Program logic error. Prog="program". Line=line_number. P1="UOW - SR3 BFOR  
hex_SR3_database_key, not found in hash table". P2=1
```

After you run the utility, restart the ECCR so that it can detect the SR2 and SR3 pairs that the utility recorded.

Running the DTLUCSR2 Utility

Run the DTLUCSR2 utility before you run the IDMS log-based ECCR the first time and after any event that tends to create SR2 and SR3 records.

Before you start the utility, ensure that you added the SR2INPUT DD statement to the IDMS log-based ECCR JCL. This DD statement points to the utility result files that contain information for building the SR2-SR3 internal table. For more information, see the *PowerExchange CDC Guide for z/OS*.

1. Edit the DTLICSRI member in the RUNLIB library.

For each database with source tables to be registered for change capture, customize the following sample statements:

```
Read,  
DD_NAME=ddname  
PAGE_GROUP=n  
RADIX=x
```

The following table describes these statements:

Statement	Description
DD_NAME	The DDNAME to be added to the DTLUCSR2 JCL. This name does not have to match a DD name from an IDMS region, but it must exactly match the DD name in the DTLUCSR2 JCL. Format: DD_NAME=STUDENT
PAGE_GROUP	If the database file is normally accessed with a page group other than zero, you must specify the PAGE_GROUP number.
RADIX	If you want to use a RADIX value other than the default of 8, enter a value from 2 to 12.

Note: DTLUCSR2 writes control information to the SR2TOTAL file and SR2/SR3 link information to the SR2OUT file. These files are created with default information at installation time. You might need to change the file sizes, depending on the number of SR3 records.

2. Add DD cards to the DTLUCSR2 JCL that match the DD names in the DTLICSRI parameter file.
The DD cards point to the relevant IDMS data set names.
3. Run the JCL in RUNLIB member DTLUCSR2.

CHAPTER 11

DTLUCUDB - DB2 for Linux, UNIX, and Windows CDC Utility

This chapter includes the following topics:

- [DTLUCUDB Utility Overview, 144](#)
- [Running the DTLUCUDB Utility, 144](#)
- [Gathering Diagnostic Information to Resolve a DB2 Capture Problem, 151](#)

DTLUCUDB Utility Overview

The DTLUCUDB utility performs the following functions:

- Creates a DB2 catalog snapshot to initialize the PowerExchange capture catalog table.
- Generates diagnostic information.

For more information about this utility, see the *PowerExchange CDC Guide for Linux, UNIX, and Windows*.

Running the DTLUCUDB Utility

You can run the DTLUCUDB utility in either of the following ways:

- Issue the command directly from the command line, for example:
- Create a file that contains the commands you want to run and then call that file from the command line, for example:

```
DTLUCUDB HELP
```

```
DTLUCUDB mycommands.txt
```

Tip: Use a file when you run a number of different commands at the same time. You can include comments in the file by prefixing the comment line with a slash and asterisk (/ *).

DTLUCUDB Utility Syntax

The DTLUCUDB syntax optionally includes database keywords for all of the command options except HELP. The database keywords provide information for connecting to a DB2 database. Although these keywords are optional, you should specify them if you do not want to use the defaults.

The following table describes the database keywords:

Keyword	Syntax	Description
DB	[DB= <i>database_name</i>]	Name of the DB2 database to which you want to connect. Default is SAMPLE.
UID	[UID= <i>user_id</i>]	User ID to use for connecting to the database. Default is logon user ID. For a database on a supported Linux, UNIX, or Windows system, if you have enabled PowerExchange LDAP user authentication and disabled relational pass-through authentication, the user ID is the enterprise user name. For more information, see the <i>PowerExchange Reference Manual</i> .
{PWD EPWD}	[{PWD= <i>password</i> EPWD= <i>encrypted_password</i> }]	Password or encrypted password for the specified user ID. Do not specify both.

In the DTLUCUDB syntax, the database keywords are represented by the italicized phrase *database keywords*.

The DTLUCUDB utility has the following syntax:

```
CCATDMP [database keywords]
  [CCATALOG=table_name]
  [FILE=file_name]
  [REPLACE={N|Y}]
;
DBINFO [database keywords]
;
DUMPDIAG [database keywords]
  [CCATALOG=table_name]
  BVTS=begin_VTS
  [EVTS=end_VTS]
  DIR=dump_directory
  [REPLACE={N|Y}]
;
HELP
;
LOGPRT [database keywords]
  [CCATALOG=table_name]
  [PART=DB partition_number]
  [FILE=file_name]
  [REPLACE={N|Y}]
  [RECSPERFILE=records_per_output_file]
  {BLSN=begin_LSN|BVTS=begin_VTS}
  [ELSN=end_LSN]
  [EVTS=end_VTS]
  [RECS=records_to_select]
  [TRANID=transaction_ID]
  [LOGICAL={Y|N}]
  [UDB={N|MIN|FMT|MAX}]
;
SETDEF [database keywords]
  [CCATALOG=table_name]
;
SNAPSHOT [database keywords]
  [CCATALOG=table_name]
  [REPLACE={N|Y}]
;
SNAPUPDT [database keywords]
  [CCATALOG=table_name]
  [REPLACE={N|Y}]
  [ARCHIVEOLDPOSITIONING={N|Y}]
;
```

```

SQUISH [database keywords]
  [CCATALOG=table_name]
  VTSDT=VTS date_time
  REPLACE={Y|N}
;
UPDTPDRP [database keywords]
  [CCATALOG=table_name]
  VTSDT={EOC|NOW|VTS date_time}
;

```

Command Options for the DTLUCUDB Utility

The DTLUCUDB utility has the following command options:

- [“CCATDMP Command” on page 146](#)
- [“DBINFO Command” on page 147](#)
- [“DUMPDIAG Command” on page 147](#)
- [“HELP Command” on page 148](#)
- [“LOGPRT Command” on page 148](#)
- [“SETDEF Command” on page 149](#)
- [“SNAPSHOT Command” on page 150](#)
- [“SNAPUPDT Command” on page 150](#)
- [“SQUISH Command” on page 150](#)
- [“UPDTPDRP Command” on page 151](#)

CCATDMP Command

The CCATDMP command produces a dump file that contains SQL insert statements corresponding to the contents of the capture catalog table.

The default file name is `ccatdmp.database_name.capture_catalog_name.sql`. The file is saved to the current working directory when the command is executed.

```

CCATDMP [database keywords]
  [CCATALOG=table_name]
  [FILE=file_name]
  [REPLACE={N|Y}]
;

```

The following table describes the parameters in the CCATDMP command:

Parameter	Description
CCATALOG	Name of the capture catalog table. Default is <code>current_user.DTLCCATALOG</code> .
FILE	Name of the dump file. This name overrides the default file name: <code>ccatdmp.database_name.capture_catalog_name.sql</code> .
REPLACE	REPLACE=Y overwrites an existing data in the file. Default is N.

DBINFO Command

The DBINFO command prints out environmental information.

```
DBINFO [database keywords];
```

An example of this type of information is:

```
PWX-20526 UDB capture DB/DBMS Info:
PWX-20527      SQL_DATABASE_NAME: CAPTURE
PWX-20527      SQL_SERVER_NAME: DB2
PWX-20527      SQL_USER_NAME: PWXUSER
PWX-20527      SQL_DBMS_NAME: DB2/NT
PWX-20527      SQL_DBMS_VER: 08.02.0004
PWX-20527 SQL_IDENTIFIER_QUOTE_CHAR: "
PWX-20527      SQL_CONNECT_CODEPAGE: 1252
PWX-20527      SQL_DATABASE_CODEPAGE: 1252
PWX-20527      SQL_APPLICATION_CODEPAGE: 1252
PWX-20527      INST_NAME: DB2
PWX-20527      IS_INST_PARTITIONABLE: 1
PWX-20527      NUM_DBPARTITIONS: 5
PWX-20527      INST_PTR_SIZE: 32
PWX-20527      RELEASE_NUM: 03050106
PWX-20527      SERVICE_LEVEL: DB2 v8.1.11.973
PWX-20527      BLD_LEVEL: s060120
PWX-20527      PTF: WR21365
PWX-20527      FIXPACK_NUM: 11
PWX-20527      OS_NAME: WIN32_NT
PWX-20527      OS_VERSION: 5.2
PWX-20527      OS_RELEASE: Service Pack 1
PWX-20527      HOST_NAME: S160019
PWX-20527      TOTAL_CPUS: 2
PWX-20527      CONFIGURED_CPUS: 4
PWX-20527      TOTAL_MEMORY: 3072
PWX-20527      CATALOG_PARTITION: 0
PWX-20528 Partition[ 0]: S160019.informatica.com, 0, S160019
PWX-20541 LSN at first DB connect: 00003921000C0000
PWX-20541 LSN at End of Log: 00003921000C0000
PWX-20528 Partition[ 1]: S160019.informatica.com, 1, S160019.informatica.com
PWX-20541 LSN at first DB connect: 0000088B800C0000
PWX-20541 LSN at End of Log: 0000088B800C0000
PWX-20506 Command DBINFO complete
```

DUMPDIAG Command

The DUMPDIAG command produces files for the capture catalog, general database information, and the DB2 log records for each partition in the directory that is specified by the DIR parameter.

```
DUMPDIAG [database keywords]
[CCATALOG=table_name]
BVTS=begin_VTS [EVTS=end_VTS]
DIR=dump_directory [REPLACE={N|Y}]
;
```

The following table describes the parameters in the DUMPDIAG command:

Parameter	Description
CCATALOG	Name of the capture catalog table.
BVTS	Required. The starting timestamp for the diagnostics file in the format YYYY-MM-DD-HH.MI.SS.SSSSSS. The year, month and day are required. BVTS values are always specified in GMT (Greenwich Mean Time).

Parameter	Description
EVTS	The ending timestamp for the diagnostics file in the format YYYY-MM-DD-HH.MI.SS.SSSSSS. The year, month and day are required. EVTS values are always specified in GMT (Greenwich Mean Time).
DIR	Required. The directory where the diagnostics file is written. The file name is "ccatdmp.database_name.capture_catalog_name.sql" and cannot be changed.
REPLACE	Indicates whether to overwrite existing files. Specify Y to overwrite existing files.

An example of this type of information is:

```
PWX-20512 Producing file 'dtst20061221\ccatdmp.cap14.partcaptst.sql'
PWX-20512 Producing file 'dtst20061221\dbconfig.txt'
PWX-20512 Producing file 'dtst20061221\p0.logdmp'
PWX-20540 Begin LSN 0000042B3EBC0000 selected for BVTS value
PWX-20519 End of UDB log file reached
PWX-20512 Producing file 'dtst20061221\p1.logdmp'
PWX-20540 Begin LSN 00000768C1040000 selected for BVTS value
PWX-20519 End of UDB log file reached
PWX-20512 Producing file 'dtst20061221\p20.logdmp'
PWX-20540 Begin LSN 0000046B76C10000 selected for BVTS value
PWX-20519 End of UDB log file reached
PWX-20506 Command DUMPDIAG complete
```

HELP Command

The HELP command prints the full syntax of the DTLUCUDB command.

LOGPRT Command

The LOGPRT command produces a file that formats the contents of the DB2 log. By default, the command creates a file named "*database_name.logprt*" in the current working directory.

The command syntax is:

```
LOGPRT [database keywords]
      [CCATALOG=table_name]
      [PART=DB partition_number]
      [FILE=file_name]
      [REPLACE={N|Y}]
      [RECSPERFILE=records_per_output_file]
      [LSN={begin_LSN|BVTS=begin_VTS}
      [ELSN=end_LSN]
      [EVTS=end_VTS]
      [RECS=records_to_select]
      [TRANID=transaction_ID]
      [LOGICAL={Y|N}]
      [UDB={N|MIN|FMT|MAX}]
;
```

The following table describes the parameters in the LOGPRT command:

Parameter	Description
CCATALOG	Name of the capture catalog table.
PART	Database partition number.

Parameter	Description
FILE	Name of the formatted log file. This overrides the default "<database name>.logprt" file.
REPLACE	Indicates whether to overwrite an existing file. Specify Y to overwrite an existing file.
RECSPERFILE	An option that can be used to divide a large amount of output into multiple files. The generated file names have the format: <i>database_name.first_lsn_value_in_file.logprt</i> . If the FILE keyword has also been specified, the generated file names have the format: <i>file_name.first_lsn_value_in_file</i> .
BLSN	A 6-byte DB2 Log Sequence Number (LSN), in hexadecimal digits, that indicates where the command is to start reading in the log. This value must represent an actual LSN. If fewer than 12 hexadecimal digits are specified, zeros are logically added to the left. BLSN defaults to the beginning of the active log. You must specify either BLSN or BVTS.
BVTS	Starting timestamp that indicates where the command is to start reading in the log. You must specify either BLSN or BVTS.
ELSN	A 6-byte DB2 Log Sequence Number (LSN), in hexadecimal digits, that specifies where the command is to stop. This value is not required to correspond to an actual LSN. If fewer than 12 hexadecimal digits are specified, zeros are logically added to the left. You can use this option to filter the output. ELSN defaults to the end of the log. You must specify either BLSN or BVTS.
EVTS	Ending timestamp that indicates where the command is to stop. You can use it to filter the output.
RECS	Number of records that indicates where the command is to stop. You can use this option to filter the output.
TRANID	Criteria for filtering output. This option does not stop the reading of log records when transaction-end log records are processed.
LOGICAL	DB2 log reading consists of reading actual DB2 log records and interpreting them into logical events (known as logical log records). The LOGICAL keyword can be used to force these log records to be printed in the file. Default is Y.
UDB	Controls how "real" DB2 log records are formatted in the file. Valid options are: <ul style="list-style-type: none"> - N. Does not print at all (default). - MIN. Prints a minimum of information. - FMT. Formats what is known about the record. - MAX. Dumps the record in hex and formats it.

SETDEF Command

The SETDEF command sets default values for keywords on the other commands.

```
SETDEF [database keywords]
      [CCATALOG=table_name]
;
```

The following table describes the parameter in the SETDEF command:

Parameter	Description
CCATALOG	Name of the capture catalog table. Default is DTLCCATALOG.

SNAPSHOT Command

The SNAPSHOT command is used to initialize capture catalog table. Note that restart points cannot precede the point in the log where a snapshot is taken. Therefore, use this command carefully.

```
SNAPSHOT [database keywords]
  [CCATALOG=table_name]
  [REPLACE={N|Y}]
;
```

The following table describes the parameters in the SNAPSHOT command:

Parameter	Description
CCATALOG	Name of the capture catalog table to be initialized. Default is DTLCCATALOG.
REPLACE	Indicates whether to overwrite any existing rows of data in the capture catalog table. If rows of data exist, you must specify Y. Default is N.

SNAPUPDT Command

Use the SNAPUPDT command after partitions are added to or dropped from the database instance. For each new partition, the command adds a new partition positioning entry in the capture catalog. For each partition that is dropped, the command removes a positioning entry from the capture catalog.

```
SNAPUPDT [database keywords]
  [CCATALOG=table_name]
  [REPLACE={N|Y}]
  [ARCHIVEOLDPOSITIONING={N|Y}]
;
```

The following table describes the parameters in the SNAPUPDT command:

Parameter	Description
CCATALOG	Name of the capture catalog table.
REPLACE	REPLACE=Y must be specified to update the capture catalog. If REPLACE is not set to Y, the command shows what changes would be made, but does not make them.
ARCHIVEOLDPOSITIONING	If you specify ARCHIVEOLDPOSITIONING=Y, the positioning entries remain in the capture catalog, but are not accessible.

SQUISH Command

Use the SQUISH command to advance the base of the capture catalog to a new VTS date and time by collapsing catalog entries (table or column alters) and removing positioning entries. Catalog (any DDL activity) and positioning (VTS, LSN, or partition set) entries are added to an active capture catalog during extraction processing.

```
SQUISH [database keywords]
  [CCATALOG=table_name]
```

```

VTSDT=VTS date_time
REPLACE={Y|N}
;

```

The following table describes the parameters in the SQUISH command:

Parameter	Description
CCATALOG	Name of the capture catalog table.
REPLACE	Specify Y to be able to update the capture catalog. If N is specified, the command shows the changes but does not make them.
VTSDT	A virtual timestamp (date and time). This timestamp value must be within the bounds of the capture catalog.

Note: Do not run the SQUISH command while extractions are active. Perform a backup before running SQUISH.

UPDTDRP Command

Use the UPDTDRP command to update the default restart point.

```

UPDTDRP [database keywords]
[CCATALOG=table_name]
VTSDT={EOC|NOW|VTS date_time}
;

```

The following table describes the parameters in the UPDTDRP command:

Parameter	Description
CCATALOG	Name of the capture catalog table.
VTSDT	Required. The value must be greater than lowest VTS value in the capture catalog and less than the current end-of-log VTS value. The value is one of the following: <ul style="list-style-type: none"> - EOC. End of catalog. - NOW. Current date and time. - VTS <i>date_time</i>. The virtual timestamp that has the specified date and time, for example, 2007-09-07.18.40.47

Gathering Diagnostic Information to Resolve a DB2 Capture Problem

Informatica Global Customer Support might request diagnostic information to use in resolving a DB2 capture problem. The following commands are example diagnostic commands that are entered at a Windows command line:

```

mkdir probl234
cd /probl234
dtlucudb dumpdiag db=mydb ccatalog=my.capturecat bvts=<start time> evts=<end time>

```

The directory, probl234, contains several files. You would zip these files and send them to Informatica Global Customer Support for analysis.

Note: If you specify the EVTS option for the DUMPDIAG command, verify that the problem section of the log is captured.

CHAPTER 12

DTLULCAT and DTLULOGC - IDMS Log Catalog Utilities

This chapter includes the following topics:

- [DTLULCAT and DTLULOGC Utilities Overview, 153](#)
- [Running the DTLULCAT Utility, 154](#)
- [Running the DTLULOGC Utility, 154](#)
- [Manually Manipulating the Log Catalog, 156](#)
- [Guidelines for Adding Logs to the Catalog with the DTLULCAT and DTLULOGC Utilities, 158](#)

DTLULCAT and DTLULOGC Utilities Overview

The Log Catalog holds information about the IDMS logs which are available for the use of PowerExchange log-based capture. During the initial installation of PowerExchange, a Log Catalog VSAM file will be created (default naming will be &HLQ..LOGSCAT) and a dummy record will be added.

For IDMS log-based capture to work effectively, it is vital to ensure that the log catalog is updated in a timely fashion and that log information is both secure and available. If the logs are not in the catalog, the records they hold will be unknown to PowerExchange. The correct way to add information to the catalog is to use utility DTLULCAT to format the input, then run DTLULOGC to amend the Log Catalog with that prepared input.

RUNLIB member DTLULCAU is supplied to run the two utilities one after the other. It is expected that this be scheduled to run as soon as the latest IDMS log had been spooled off. There may, however, be times when DTLULOGC is run in isolation, involving manual coding of the input file.

Correct scheduling of the addition logs to the Log Catalog is vital to obtaining timely data from the log-based IDMS capture environment.

RELATED TOPICS:

- [“Guidelines for Adding Logs to the Catalog with the DTLULCAT and DTLULOGC Utilities” on page 158](#)

Running the DTLULCAT Utility

This utility program is used to take the supplied journal name and use it to prepare the input required by the catalog utility program DTLULOGC. The utility is delivered as an executable on Windows and member DTLULCAT in RUNLIB on MVS.

Sample statements follow:

```
IDMS_VERSION=15
FILE_TYPE=C
MEDIA_TYPE=D
MEDIA_CONTENT=BI
SERVICE=IDMSE150
INSTANCE_IDENTIFIER=XYLOGSID
```

The following table describes the sample statements:

Parameter	Description
IDMS_VERSION	The version in which the journal record structure last changed. The value of 15 is the only valid value for the supported IDMS versions, all of which are later than IDMS Version 14.
FILE_TYPE	File type. Specifies one of the following: <ul style="list-style-type: none">- C. Central version.- L. Local mode.
MEDIA_TYPE	Specifies one of the following: <ul style="list-style-type: none">- T. Tape.- D. Disk.
MEDIA_CONTENT	Determines the images of changed records delivered: <ul style="list-style-type: none">- BI. Before images.- AI. After images.- BA. Both before and after images.
SERVICE	IDMS CV name or Local Job name.
INSTANCE_IDENTIFIER	Chosen LOGSID identifier.

The utility DTLULCAT writes to DDCARD SYSPUNCH. This file is then the input to utility DTLULOGC.

Running the DTLULOGC Utility

The utility DTLULOGC populates the log catalog with information about the logs to process. The example below shows sample JCL DTLULCAU to run both DTLULCAT followed by DTLULOGC. Running DTLULCAU JCL is the recommended method of adding to the Log Catalog.

This example adds log DTLUSR.IDMS.E15SP0.OFF.LOADED.JOURNAL1 for an IDMS Version 18 environment with CV Name IDMSE150, where the log resides on disk storage and will be accessed using a LOGSID value of XYLOGSID. Here the SYSIN data is shown as instream for clarity, but the sample JCL is delivered pointing to member DTLIDLC when running against a CV (DTLIDLL for Local Job mode) in which these statements would normally be placed.

```

/*****
/*
/* SAMPLE JCL TO:-
/*
/* CAPTURE IDMS JOURNAL FILE INFORMATION AND INPUT STREAM
/* INTO FOR DTLULOGC LOG FILE CATALOG ROUTINE.
/*
/* NORMALLY THE SYSIN INPUT STREAM WOULD BE A PDS MEMBER.
/*
/* THIS NEEDS TO BE INTEGRATED INTO THE END USERS JOURNAL
/* ARCHIVING PROCEDURE, WHICH MAY BE DIFFERENT FROM SITE TO SITE.
/*
/* A MECHANISM WILL NEED TO BE ESTABLISHED TO REPLACE THE DATASET
/* SPECIFIED VIA THE LOGFILE DD STATEMENT WITH THE LOGFILE
/* WHICH IS CURRENTLY THE OBJECT OF THE USERS ARCHIVING PROCEDURE
/* AND OUR CATALOG OPERATION
/*
/*****
//INCS1 INCLUDE MEMBER=GENBULK
//DTLULCAT EXEC PGM=DTLULCAT
//STEPLIB DD DISP=SHR,DSN=DTLUSR.V800B14.LOADLIB
//DTLCFG DD DISP=SHR,DSN=DTLUSR.V800B14.RUNLIB(DBMOVER)
//DTLKEY DD DISP=SHR,DSN=DTLUSR.V800B14.RUNLIB(LICENSE)
//DTLMSG DD DISP=SHR,DSN=&HLQ..DTLMSG,FREE=CLOSE
//DTLLOG DD SYSOUT=*
//LOGFILE DD DISP=SHR,DSN=DTLUSR.IDMS.E15SP0.OFF.LOADED.JOURNAL1
//SYSPRINT DD SYSOUT=*
//SYSPUNCH DD DSN=&&LOGDATA,
// DISP=(,PASS),
// SPACE=(CYL,(2,1),RLSE),
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSIN DD *
IDMS_VERSION=15
FILE_TYPE=C
MEDIA_TYPE=D
MEDIA_CONTENT=BI
SERVICE=IDMSE150
INSTANCE_IDENTIFIER=XYLOGSID
/*
//DTLULOGC EXEC PGM=DTLULOGC
//STEPLIB DD DISP=SHR,DSN=DTLUSR.V800B14.LOADLIB
//DTLCFG DD DISP=SHR,DSN=DTLUSR.V800B14.RUNLIB(DBMOVER)
//DTLKEY DD DISP=SHR,DSN=DTLUSR.V800B14.RUNLIB(LICENSE)
//DTLSGN DD DISP=SHR,DSN=DTLUSR.V800B14.RUNLIB(SIGNON)
//DTLMSG DD DISP=SHR,DSN=&HLQ..DTLMSG
//LOGSCAT DD DISP=SHR,DSN=DTLUSR.V800B14.V1.LOGSCAT
//DTLLOG DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//REPORT DD SYSOUT=*
//EXPORT DD SYSOUT=*
//SYSIN DD DISP=SHR,DSN=&&LOGDATA

```

Note: For IDMS_VERSION, specify 15 for IDMS versions later than version 14.

Manually Manipulating the Log Catalog

During the normal course of IDMS log processing, the Log Catalog will be updated using the combination of DTLULCAT and DTLULOGC to add the next available log. You might need to alter details for log entries or remove logs from the catalog. To do this, DTLULOGC (DTLULOGC JCL in RUNLIB) will be run stand-alone with hand-coded input.

The utility allows the user to:

- Add an instance
- Add a log
- Update a log entry
- Delete an entry
- Export an entry to another data set for offload

The following list shows the keywords and parameters available to code in an 80 byte file, which you specify as input in the SYSIN DD card. See the sample JCL.

ADD_INSTANCE parameters

Add a LOGSID instance to the catalog. Each LOGSID used requires an instance to be added to the log catalog.

The following table shows the parameters available for the ADD_INSTANCE keyword:

Parameter	Description
INSTANCE_IDENTIFIER	LOGSID value
VERSION	Version number of the entry

ADD_ENTRY parameters

Adds a specific log to the log catalog.

The following table shows the parameters available for the ADD_ENTRY keyword:

Parameter	Description
BLOCK_SIZE	Block size of the log. Required if the logs are to be shipped to another platform.
ENTRY_NUMBER	Sequential number, which should be incremented by 1 for each new log added to the log catalog.
FILE_TYPE	<ul style="list-style-type: none">- C. Central or Shared Service Log or Journal.- L. Local Mode or Unshared Service Log or Journal.
FIRST_RECORD_SEQUENCE_NUMBER	Sequence number of the first record in the block.
FIRST_RECORD_TIME_STAMP	Timestamp of the first record in the block.
IDMS_VERSION	Version number of IDMS. Specified as an integer.
INSTANCE_IDENTIFIER	LOGSID value

Parameter	Description
LAST_RECORD_IDENTIFIER	Record ID of the last record in the block or zeros if a non-data record.
LAST_RECORD_OFFSET	Offset of last valid offset in the block.
LOG_DATA_TYPE	IDL for MVS IDMS log data.
LOG_FILE_NAME	Name of IDMS log file.
MEDIA_CONTENT	<ul style="list-style-type: none"> - AI. Only contains After images. - BI. Only contains Before images. - BA. Contains both Before and After images.
MEDIA_TYPE	<ul style="list-style-type: none"> - D. Disk. - T. Tape.
NUMBER_OF_BLOCKS	Number of blocks in the log.
SERVICE	CV name or Local Mode job name.
STATUS	<ul style="list-style-type: none"> - A. Active. - S. Skip. - T. Terminate.
ENTRY_TYPE	<ul style="list-style-type: none"> - 1. File entry. - 2. Reserved for future use.
VERSION	Version number of the entry.

UPDATE_ENTRY parameters

Updates a log entry. The entry is identified by the value of INSTANCE_IDENTIFIER and ENTRY_NUMBER.

Valid parameters are those listed for ADD_ENTRY.

DELETE_ENTRY INSTANCE_IDENTIFIER=instance_identifier

Deletes the oldest log for the specified INSTANCE_IDENTIFIER.

REPORT_INSTANCE INSTANCE_IDENTIFIER=instance_identifier

Lists catalog entries for the specified INSTANCE_IDENTIFIER.

EXPORT_INSTANCE INSTANCE_IDENTIFIER=instance_identifier

Used to export all information for a specified INSTANCE_IDENTIFIER to a file.

Note: Keyword commands are separated by a semi-colon (;), parameters by a comma (,).

The following sample input adds two instances (LOGSIDs), adds entries (log files), deletes an entry, reports instance LOGSIDA, exports instance LOGSIDA to a file (dtlulgc.txt), and finally deletes instance LOGSIDA:

```
ADD_INSTANCE INSTANCE_IDENTIFIER=LOGSIDA, VERSION=224;
ADD_ENTRY INSTANCE_IDENTIFIER=LOGSIDA, ENTRY_NUMBER=777, VERSION=0, ENTRY_TYPE=1,
STATUS=A, LOG_DATA_TYPE=IDL, IDMS_VERSION=15, FILE_TYPE=C, MEDIA_TYPE=D,
MEDIA_CONTENT=BI, SERVICE=IDMSE150, LOG_FILE_NAME=XXXXXXXXXXXXXXXXXXXXXXXXXXXX,
BLOCK_SIZE=29000, NUMBER_OF_BLOCKS=445, LAST_RECORD_OFFSET=1119,
LAST_RECORD_IDENTIFIER=3, FIRST_RECORD_SEQUENCE_NUMBER=4,
FIRST_RECORD_TIME_STAMP="05/03/03 10:55:01";
ADD_ENTRY INSTANCE_IDENTIFIER=LOGSIDA, ENTRY_NUMBER=778, VERSION=0, ENTRY_TYPE=1,
STATUS=A, LOG_DATA_TYPE=IDL, IDMS_VERSION=15, FILE_TYPE=C, MEDIA_TYPE=D,
```

```
MEDIA_CONTENT=BI, SERVICE=IDMSEI50, LOG_FILE_NAME=MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM,
BLOCK_SIZE=29000, NUMBER_OF_BLOCKS=445, LAST_RECORD_OFFSET=1119,
LAST_RECORD_IDENTIFIER=3, FIRST_RECORD_SEQUENCE_NUMBER=4,
FIRST_RECORD_TIME_STAMP="05/03/03 12:55:01";
ADD_ENTRY INSTANCE_IDENTIFIER=LOGSIDA, ENTRY_NUMBER=779, VERSION=0, ENTRY_TYPE=1,
STATUS=A, LOG_DATA_TYPE=IDL, IDMS_VERSION=15, FILE_TYPE=C, MEDIA_TYPE=D,
MEDIA_CONTENT=BI, SERVICE=IDMSEI50, LOG_FILE_NAME=ZZZZZZZZZZZZZZZZZZZZCCCCCCCCCC,
BLOCK_SIZE=29000, NUMBER_OF_BLOCKS=333, LAST_RECORD_OFFSET=1119,
LAST_RECORD_IDENTIFIER=3, FIRST_RECORD_SEQUENCE_NUMBER=4,
FIRST_RECORD_TIME_STAMP="05/03/03 14:55:01";
ADD_INSTANCE INSTANCE_IDENTIFIER=ABCDE, VERSION=0;
ADD_ENTRY INSTANCE_IDENTIFIER=ABCDE, ENTRY_NUMBER=1, VERSION=0, ENTRY_TYPE=1, STATUS=A,
LOG_DATA_TYPE=IDL, IDMS_VERSION=15, FILE_TYPE=C, MEDIA_TYPE=D, MEDIA_CONTENT=BI,
SERVICE=IDMSEI5P, LOG_FILE_NAME=BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB, BLOCK_SIZE=29000,
NUMBER_OF_BLOCKS=444, LAST_RECORD_OFFSET=1112, LAST_RECORD_IDENTIFIER=2,
FIRST_RECORD_SEQUENCE_NUMBER=3, FIRST_RECORD_TIME_STAMP="05/04/03 08:55:01";
ADD_ENTRY INSTANCE_IDENTIFIER=ABCDE, ENTRY_NUMBER=2, VERSION=0, ENTRY_TYPE=1, STATUS=A,
LOG_DATA_TYPE=IDL, IDMS_VERSION=15, FILE_TYPE=C, MEDIA_TYPE=D, MEDIA_CONTENT=BI,
SERVICE=IDMSEI5P, LOG_FILE_NAME=CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC, BLOCK_SIZE=29000,
NUMBER_OF_BLOCKS=445, LAST_RECORD_OFFSET=1119, LAST_RECORD_IDENTIFIER=3,
FIRST_RECORD_SEQUENCE_NUMBER=4, FIRST_RECORD_TIME_STAMP="05/04/03 10:55:01";
UPDATE_ENTRY INSTANCE_IDENTIFIER=LOGSIDA, ENTRY_NUMBER=779, VERSION=0, ENTRY_TYPE=1,
STATUS=A, LOG_DATA_TYPE=IDL, IDMS_VERSION=15, FILE_TYPE=C, MEDIA_TYPE=D,
MEDIA_CONTENT=BI, SERVICE=DTLXXXXX, LOG_FILE_NAME=AAAAAAAAAAAAAAKKKKKKKKKKKKKKK,
BLOCK_SIZE=29000, NUMBER_OF_BLOCKS=111, LAST_RECORD_OFFSET=1119,
LAST_RECORD_IDENTIFIER=3, FIRST_RECORD_SEQUENCE_NUMBER=4,
FIRST_RECORD_TIME_STAMP="05/04/03 12:55:01";
DELETE_ENTRY INSTANCE_IDENTIFIER=LOGSIDA;
REPORT_INSTANCE INSTANCE_IDENTIFIER=LOGSIDA;
EXPORT_INSTANCE INSTANCE_IDENTIFIER=LOGSIDA;
DELETE_INSTANCE INSTANCE_IDENTIFIER=LOGSIDA;
```

- Care must be taken with the order in which the logs are added to the catalog. Operational procedures for the running of DTLULCAT and DTLULOGC must be developed to ensure that logs are added in the correct sequence.

The rule used for checking such log additions is:

- A local mode journal must not be added to the catalog if the last available timestamp within the journal is greater than the timestamp of the previously added CV mode journal.
- If logs are added in the incorrect sequence expect to see messages similar to the following:

[illegible]

CHAPTER 13

DTLURDMO - Data Map Utility

This chapter includes the following topics:

- [DTLURDMO Utility Overview, 160](#)
- [Supported Operating Systems for the DTLURDMO Utility, 161](#)
- [Control Statement Overview for the DTLURDMO Utility, 161](#)
- [Control Statement Syntax for the DTLURDMO Utility, 161](#)
- [Control Statements and Parameters for the DTLURDMO Utility , 162](#)
- [Scope of Operands, 191](#)
- [Running the DTLURDMO Utility on i5/OS, 193](#)
- [Running the DTLURDMO Utility on Linux, UNIX, and Windows, 193](#)
- [Running the DTLURDMO Utility on z/OS, 194](#)
- [DTLURDMO Control Statement Examples, 194](#)
- [DTLURDMO Report Examples, 199](#)

DTLURDMO Utility Overview

Use the DTLURDMO utility to copy the following types of definitions from one environment or location to another:

- PowerExchange data maps
- PowerExchange capture registrations
- PowerExchange extraction maps

When performing a copy, you can optionally change certain attributes of the new capture registration, data map, or extraction map, such as the schema name or table name. The DTLURDMO utility can be used to promote maps and registrations between software environments or versions. It can also be used as part of a software change management system.

Supported Operating Systems for the DTLURDMO Utility

You can run the DTLURDMO utility on the following platforms:

- i5/OS
- Linux, UNIX, and Windows
- z/OS

Control Statement Overview for the DTLURDMO Utility

DTLURDMO control statements are of the following types:

- Global statements control overall program execution or provide basic information, such as user name or password. Global statements remain active for the entire DTLURDMO execution. You can include them only once in the input file or stream. The REPORTDEST global statement should be entered at the beginning of the statement set. You can include multiple REPORTDEST statements in the input file or stream. The remaining statements can be included only once in the input.
- Copy statements specify the type of copy to perform:
 - DM_COPY copies data maps.
 - REG_COPY copies capture registrations and, optionally, extraction maps.
 - XM_COPY copies extraction maps.Copy statements have no operands but can be followed by optional statements. Only a single type of copy statement can appear in the input file or stream, but it can appear multiple times.
- Optional statements follow a copy statement and are valid only for the scope of the execution of the copy statement. Optional statements become inactive when PowerExchange encounters a subsequent copy statement. Optional statements filter the objects selected, rename objects, change object attributes, and set optional functions for the copy.

Control Statement Syntax for the DTLURDMO Utility

The DTLURDMO definition file includes the following control statements:

```
[REPORTDEST LOG|STDERR|STDOUT|file_name];
[INPUT data_set_name|file_name|folder_name;]
[OUTPUT data_set_name|file_name|folder_name;]
USER user_ID;
[PWD password|EPWD encrypted_password];
[TARGETUSER target_user_ID;]
[TARGETPWD password|TARGETEPWD encrypted_password;]
SOURCE source_node;
TARGET target_node;
[REPLACE;]
[DETAIL;]
[NOTIMESTAMPS;]
```

```
[TESTMODE;]
[VALIDATE;]
[DM_COPY;
[DM_COPY_optional_statements;]]
[REG_COPY;
[REG_COPY_optional_statements;]]
[XM_COPY;
[XM_COPY_optional_statements;]]
```

In the syntax, statements or parameters enclosed in square brackets ([]) are optional. The following rules and guidelines apply:

- All control statements must end with a semicolon (;).
- Statements and parameters are case-insensitive.
- Operands used for comparison, such as operands used to filter objects, are case-insensitive.
- Operands used to rename or modify object attributes are case-sensitive.
- Parameters that are enclosed in parentheses and comma-separated must be specified in that format.
- You must specify exactly one PWD or EPWD statement.
- The INPUT statement applies to the SOURCE location.
- The OUTPUT statement applies to the TARGET location.
- You must specify exactly one type of copy statement: DM_COPY, REG_COPY, or XM_COPY. You can specify this statement once or multiple times.
- Optional statements follow a copy statement and are valid only for the scope of the execution of the copy statement.

Within the scope of a copy statement, DTLURDMO allows multiple occurrences of each of the EXCLUDE, MODIFY, RENAME, and SELECT statements. However, in most cases it is preferable to include multiple copy statements instead. For more information, see [“Scope of Operands” on page 191](#).

Note: Before you use a DTLURDMO definition file that you used with a previous release of the product, verify that its syntax is consistent with the syntax described in this topic.

RELATED TOPICS:

- [“Control Statements and Parameters for the DTLURDMO Utility ” on page 162](#)

Control Statements and Parameters for the DTLURDMO Utility

This section describes the control statements and their parameters. The section is organized as follows:

- Global statements
- DM_COPY statement
- REG_COPY statement
- XM_COPY statement

The discussion of each copy statement includes a description of its optional statements and their parameters.

Global Statements

Global statements remain active for the entire DTLURDMO execution. You can include multiple REPORTDEST statements in the input file or stream. The remaining statements can be included only once in the input. The following DTLURDMO statements are global:

- DETAIL
- EPWD
- INPUT
- NOTIMESTAMPS
- OUTPUT
- PWD
- REPLACE
- REPORTDEST
- SOURCE
- TARGET
- TARGETEPWD
- TARGETPWD
- TARGETUSER
- TESTMODE
- USER
- VALIDATE

DETAIL Statement

The DETAIL statement causes DTLURDMO to print a detailed report containing information about the copying process including all changes made and renames performed.

The DETAIL statement has no operands.

This statement is optional.

EPWD Statement

The EPWD statement specifies an encrypted password for the user ID specified in the USER statement. The password and user ID are used to access maps and registrations on the source system.

If the maps and registrations are on an i5/OS or z/OS system, you can enter an encrypted PowerExchange passphrase instead of an encrypted password. Ensure that the passphrase you encrypt does not contain invalid characters, such as double-quotation marks, single quotation marks, or currency symbols.

If the TARGETEPWD statement is not specified, the EPWD statement also specifies the encrypted password or passphrase for the target to which maps and registrations are copied.

Use the following syntax:

```
EPWD encrypted_password_or_passphrase;
```

Note: To encrypt a password or passphrase, you can use the **File > Encrypt** command in the PowerExchange Navigator.

Do not also specify the PWD statement.

INPUT Statement

The INPUT statement specifies an alternative input data set, file, or directory for the input maps or registrations on the source node.

Specify one of the following options:

- For DM_COPY or XM_COPY commands, a VSAM KSDS data set if the source node is on z/OS, a partitioned file if the source node is on IBM i, or a directory if the source node is on Linux, UNIX, or Windows.
- For REG_COPY commands, a CCT file name.

Use the following syntax:

```
INPUT data_set_name|file_name|folder_name;
```

This statement is optional. If you do not include the INPUT statement, the default folder, data set, or file for the source node is used.

NOTIMESTAMPS Statement

Specify the NOTIMESTAMPS statement to suppress timestamps in the report output.

This statement is optional. It is primarily used for testing purposes. It suppresses the timestamp on the first line of the output, so reports run at different times can be more easily compared.

OUTPUT Statement

The OUTPUT statement specifies an alternative destination data set, file, or directory for the output maps or registrations from the default destination that is specified in the DBMOVER configuration file.

Specify one of the following options:

- For DM_COPY or XM_COPY commands, a VSAM KSDS data set if the source node is on z/OS, a partitioned file if the source node is on IBM i, or a directory if the source node is on Linux, UNIX, or Windows.
- For REG_COPY commands, a CCT file name.

Use the following syntax:

```
OUTPUT data_set_name|file_name|folder_name;
```

This statement is optional.

Caution: Do not use the OUTPUT statement with the REG_COPY statement if you also use the CREATEXMAPS statement.

PWD Statement

The PWD statement specifies a password for the user ID specified in the USER statement. The password and user ID are used to access maps and registrations on the source system.

If the maps and registrations are on an i5/OS or z/OS system, you can enter a valid PowerExchange passphrase instead of a password.

If the TARGETPWD statement is not specified, the PWD statement also specifies the password or passphrase for the target to which maps and registrations are copied.

Use the following syntax:

```
PWD password_or_passphrase;
```

Use a PowerExchange passphrase for enhanced security. An i5/OS passphrase can be from 9 to 31 characters in length. A z/OS passphrase can be from 9 to 128 characters in length. A passphrase can contain the following characters:

- Uppercase and lowercase letters
- The numbers 0 to 9
- Spaces
- The following special characters:

' - ; # \ , . / ! % & * () _ + { } : @ | < > ?

Note: The first character is an apostrophe.

Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.

If a passphrase contains spaces, you must enclose it with double-quotation marks ("), for example, "This is a passphrase". If a passphrase contains special characters, you must enclose it with triple double-quotation characters ("""), for example, """"This passphrase contains special characters ! % & *. """". If a passphrase contains only alphanumeric characters without spaces, you can enter it without delimiters.

To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER configuration member. On z/OS, the DTLCFG DD statement in the DTLURDMO JCL points to the DBMOVER member. On i5/OS, the DBMOVER member is the CFG file in the PowerExchange *datalib* library. For more information, see "SECURITY Statement" in the *PowerExchange Reference Manual*.

Note: On z/OS, a valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.

Do not also specify the EPWD statement.

REPLACE Statement

The REPLACE statement causes any existing maps or registrations at the target destination to be overwritten by those copied. You can use this statement to change map or registration attributes by copying from and to the same location. To ensure that you do not inadvertently lose existing map or registration information, use this statement with caution.

The REPLACE statement has no operands.

This statement is optional.

REPORTDEST Statement

The REPORTDEST statement specifies one or more destinations for the report that you generate when you run the utility.

You can use this statement to send the report to a standard output location or to a separate file on the computer where the utility runs. You can specify up to four report locations by including multiple REPORTDEST statements in the utility input. As a best practice, include REPORTDEST statements at the beginning of the command.

Use the following syntax:

```
REPORTDEST LOG|STDERR|STDOUT|file_name
```

The REPORTDEST statement accepts the following options:

LOG

Send report output to the PowerExchange log (detail.log). The report prefixes each line in the log with a time stamp and process ID.

STDERR

Send report output to the stderr file. This option can be used on Linux, UNIX, and Windows to pipe the output to a file in addition to stdout, as shown in the second example.

STDOUT

Send report output to the stdout file. Use this option on Linux, UNIX, and Windows to view the utility progress a command window.

file_name

Send the output to a file on the local computer. Specify this option to write a permanent copy of the report to a file.

Note: If you specify REPORTDEST statements with both the STDERR and STDOUT options, and you do not use piping to direct the STDERR output to a file, the utility writes each report line twice in the command window.

Default is **LOG**.

Examples:

If you run DTLURDMO Linux, Unix, or Windows, you can review the utility progress in a command window and write the report to a file for later review. The following example shows how to specify those locations:

```
REPORTDEST stdout;  
REPORTDEST c:\reports\MyDTLURDMO_ini_report.txt;
```

If you run DTLURDMO on Linux, Unix, or Windows, you can redirect the output to a file. The following example shows how to use piping to specify a report location.

In the input file C:\INIS\MYDTLURDMO_ini, specify the following report destinations:

```
REPORTDEST stderr;
```

When you run the utility, use the following syntax to redirect the output to the MyDTLURDMO_ini_report.txt file:

```
DTLURDMO.EXE C:\INIS\MyDTLURDMO_ini >c:\reports\MyDTLURDMO_ini_report.txt;
```

SOURCE Statement

The SOURCE statement specifies the source node that contains the maps and registrations. This node must be specified in a NODE statement in the DBMOVER configuration file.

Use the following syntax:

```
SOURCE source_node;
```

This statement is optional. The default value is local.

TARGET Statement

The TARGET statement specifies the target node to which you want to copy the maps and registrations. This node must be specified in a NODE statement in the DBMOVER configuration file.

Use the following syntax:

```
TARGET target_node;
```

This statement is optional on Linux, UNIX, and Windows. The default value is local. You can specify the same node name in the SOURCE and TARGET statements.

Caution: On z/OS systems, always specify a target node in the TARGET statement so that the PowerExchange Listener writes to the target files. If you use the default value of local, intermittent file contention might occur because the DTLURDMO job might try to write to the file when the PowerExchange Listener already has the

file in write mode. If the PowerExchange Listener always writes to the target files, no such contention can occur.

TARGETEPWD Statement

The TARGETEPWD statement specifies an encrypted password for the user ID specified in the USER statement. The encrypted password and user ID are used to access maps and registrations on the target system.

If the maps and registrations are on an i5/OS or z/OS system, you can enter an encrypted PowerExchange passphrase instead of an encrypted password. Ensure that the passphrase you encrypt does not contain invalid characters, such as double-quotation marks, single quotation marks, or currency symbols.

Use the following syntax:

```
TARGETEPWD encrypted_password_or_passphrase;
```

Note: To encrypt a password or passphrase, you can use the **File > Encrypt** command in the PowerExchange Navigator.

If the DTLURDMO definition file includes the TARGETUSER statement but does not include either the TARGETEPWD or TARGETPWD statement, the EPWD or PWD statement specifies the password on the target system.

Do not also specify the TARGETPWD statement.

TARGETPWD Statement

The TARGETPWD statement specifies a password for the user ID in the TARGETUSER statement. The password and user ID are used to access maps and registrations on the target system.

If the maps and registrations are on an i5/OS or z/OS system, you can enter a valid PowerExchange passphrase instead of a password.

Use the following syntax:

```
TARGETPWD target_password_or_passphrase;
```

Use a PowerExchange passphrase for enhanced security. An i5/OS passphrase can be from 9 to 31 characters in length. A z/OS passphrase can be from 9 to 128 characters in length. A passphrase can contain the following characters:

- Uppercase and lowercase letters
- The numbers 0 to 9
- Spaces
- The following special characters:

' - ; # \ , . / ! % & * () _ + { } : @ | < > ?

Note: The first character is an apostrophe.

Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.

If a passphrase contains spaces, you must enclose it with double-quotation marks ("), for example, "This is a passphrase". If a passphrase contains special characters, you must enclose it with triple double-quotation characters ("""), for example, """This passphrase contains special characters ! % & *.""". If a passphrase contains only alphanumeric characters without spaces, you can enter it without delimiters.

To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER configuration member. On z/OS, the DTLCFG DD statement in the DTLURDMO JCL

points to the DBMOVER member. On i5/OS, the DBMOVER member is in the CFG file in the PowerExchange *datalib* library. For more information, see "SECURITY Statement" in the *PowerExchange Reference Manual*.

Note: On z/OS, a valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.

If the DTLURDMO definition file includes the TARGETUSER statement but does not include either the TARGETEPWD or TARGETPWD statement, the EPWD or PWD statement specifies the password on the target system.

Do not also specify the TARGETEPWD statement.

TARGETUSER Statement

The TARGETUSER statement specifies the user ID for accessing maps and registrations on the target system.

Use the following syntax:

```
TARGETUSER user_ID;
```

This statement is optional. If it is not specified, the USER statement specifies the user ID for both the source and target systems.

TESTMODE Statement

Use the TESTMODE statement to generate a report of the maps or registrations selected for processing without making any in-memory changes or updating the output files.

The TESTMODE statement tests the effect of EXCLUDE and SELECT statements so that copy scenarios can be modeled and refined.

The TESTMODE statement produces a report with the following information:

- The maps or registrations included in the initial result set.
- The maps or registrations that are excluded by the EXCLUDE parameter. The EXCLUDE parameter takes precedence over the SELECT parameter.
- The maps or registrations that are selected by the SELECT parameter.
- The total number of maps or registrations read by the utility.

This statement is optional, and has no operands.

Note: If you run the utility with both TESTMODE and VALIDATE statements in the input, the utility runs the TESTMODE statement. If you run the utility without the TESTMODE statement or VALIDATE statement, the utility runs in update mode.

USER Statement

The USER statement specifies a user ID for accessing maps and registrations on the source system.

If you have enabled PowerExchange LDAP user authentication on a supported Linux, UNIX, or Windows source system, the user name is the enterprise user name. For more information, see the *PowerExchange Reference Manual*.

If the DTLURDMO definition file does not include a TARGETUSER statement, the USER statement also specifies the user ID on the target system.

Use the following syntax:

```
USER user_ID;
```


This statement is required. You must also specify either the PWD or EPWD statement.

VALIDATE Statement

Use the VALIDATE statement to test selection steps and modifications of maps and registrations in memory, without updating the output files.

The VALIDATE statement generates a report with similar information to the TESTMODE report and the following additional information:

- The error messages that are encountered during the validation process.
- The maps or registrations that are changed by RENAME or MODIFY parameters.
- The total number of attributes that are changed in memory but not committed.

When you run DTLURDMO in VALIDATE mode, the utility does not validate target connection information. For example, DTLURDMO does not test for valid node names, version mismatches, or target object authority.

This statement is optional and has no operands.

Note: If you run the utility with both TESTMODE and VALIDATE statements in the input, the utility runs the TESTMODE statement. If you run the utility without the TESTMODE statement or VALIDATE statement, the utility runs in update mode.

DM_COPY Statement

The DM_COPY statement copies data maps from the source to the target system. This statement has no operands but can be followed by other statements that filter the selected objects, rename objects, or change object attributes.

The following table summarizes the optional statements that can follow the DM_COPY statement:

Optional Statement	Parameters
[EXCLUDE]	[AM=access_method] [MAP=map_name] [SCHEMA=schema_name]
[MODIFY]	AM=access_method [DB2INSTANCE=db2_instance] [DB2TABLE=db2_table_name] [DBD=dbd_name] [DBID=database_ID] [DBNAME=database_name] [DDSNODENAME=ddsnode_name] [DICTNAME=dictionary_name] [FILEID=file_ID] [FN=file_name] [IMSID=ims_ID] [IMSTYPE=imstype] [MUFNAME=muf_name] [PCB=pcb_name] [PCBNUM=pcb_number] [PROGNAME=program_name] [PSB=psb_name] [SUBSCHEMA=subschema_name]

Optional Statement	Parameters
[RENAME]	[DBD=(old_dbd_name,new_dbd_name)] [IDMS_PAGEGROUP_RADIX=(old_page_group,new_page_group,old_radix,new_radix)] [MAP=(old_map_name,new_map_name)] [SCHEMA=(old_schema_name,new_schema_name)] [TABLE=(old_table_name,new_table_name)]
[SELECT]	[AM=access_method] [MAP=map_name] [SCHEMA=schema_name]

DM_COPY EXCLUDE Statement

The EXCLUDE statement specifies filter criteria for data maps to be explicitly excluded from the copying process.

The EXCLUDE statement is optional. By default, no items are excluded.

The EXCLUDE statement has the following parameters:

AM=access_method

Specifies the access method of the data maps to exclude.

MAP=map_name

Specifies a map name to exclude.

SCHEMA=schema_name

Specifies a schema name to exclude.

The parameters for the EXCLUDE statement are the same as those for the SELECT statement.

RELATED TOPICS:

- [“DM_COPY SELECT Statement” on page 174](#)

DM_COPY MODIFY Statement

The MODIFY statement modifies attributes of the copied data maps.

The AM= parameter is required with the MODIFY statement. Depending on the value that you specify for the AM= parameter, you can include additional parameters.

The MODIFY statement has the following parameters:

AM=access_method

Required. Specifies the access method to which the MODIFY statement applies.

The following table shows the additional parameters that are available for each access method:

Access Method	Available Parameters
ADABAS	DBID, FILEID
DB2	DB2TABLE, DB2INSTANCE
DB2UDB	DB2TABLE, DB2INSTANCE
DB2UNLD	FN, DB2TABLE, DB2INSTANCE
DCOM	DBID, DBNAME, MUFNAME
DL1	DBD, IMSID, IMSTYPE, PCBNUM
ESDS	FN
IDMS	SUBSCHEMA, DBNAME, PROGNAME, DICTNAME, DDSNODENAME
IMS	None
KSDS	FN
MSSQL	None
ODBA	DBD, IMSID, IMSTYPE, PSB, PCB
Oracle	None
RRDS	FN
SEQ	FN

DB2TABLE=db2_schema.db2_table_name

Modifies the DB2 schema or table name for the table mapped by the data map. For example:

```
DB2TABLE=DSN8910.EMP
```

DB2INSTANCE=db2_instance

Modifies the DB2 database name or subsystem ID for the data mapped by the data map. For example:

```
DB2INSTANCE=sample
```

DBD=dbd_name

Modifies the DBD name for the data mapped by the data map. For example:

```
DBD=PROD001
```

The DBD parameter of the DM_COPY MODIFY statement is deprecated but supported for backward compatibility. Instead, use the DM_COPY RENAME statement with the DBD parameter to rename a DBD in a data map.

DBID=database_ID

Specifies a new database ID to be used when reading the database.

DBNAME=database_name

Specifies a new database name to be used when reading the database.

DDSNODENAME=ddsnode_name

Specifies a new DDSNODE name to be used when reading the IDMS database.

DICTNAME=dictionary_name

Specifies a new dictionary name to be used when reading the IDMS database.

FILEID=file_ID

Specifies a new ADABAS file ID for the data mapped by the data map.

FN=filename

Specifies a new name for the data file associated with the data map. The file name must be a valid file name on the target system. The following examples specify a new data file name for z/OS, VSAM, and Windows, respectively:

```
FN='DATA01.SEQ.FILE'  
FN='SYS01.KSDS.DATA.FILE'  
FN=c:\myfolder\myfile.txt
```

IMSID=ims_ID

Specifies a new IMS system ID for the data mapped by the data map. For example:

```
IMSID=IMS7
```

IMSTYPE=ims_type

Specifies a new IMS type for the data mapped by the data map. Specify one of the following values:

- DEDB
- GSAM
- HDAM
- HIDAM
- HSAM
- HISAM
- MSDB
- PHDAM
- PHIDAM
- PSINDEX
- SHISAM

MUFNAME=muf_name

Specifies a new MUF name to be used when reading the Datacom database.

PCB=pcb_name

Specifies a new PCB name for the data mapped by the data map. For example:

```
PCB=PCB020
```

PCBNUM=pcb_number

Specifies a new PCB number for the data mapped by the data map.

PROGNAME=program_name

Specifies a new program name to be used when reading the IDMS database.

PSB=psb_name

Specifies a new PSB for the data mapped by the data map. For example:

```
PSB=DTL003
```

SUBSCHEMA=subschema_name

Specifies a new subschema name to be used when reading the IDMS database.

DM_COPY RENAME Statement

The RENAME statement specifies which elements of the data map name are renamed on the target system. For most parameters, the first operand represents the item or items being renamed, and the second operand represents the new name. For example:

```
MAP=(map01,map02)
```

The first operand can be any of the following:

- The full name of the item being renamed
- A partial name with a wildcard (*)
- A wildcard (*) only

In each case, all items matching the specified name or pattern are renamed to the value of the second operand.

DBD=(old_dbd_name,new_dbd_name)

For IMS DL1 or ODBA data maps, renames the DBDs in the data map that match the name or pattern in the first operand to the specified name.

In the following example, all DBDs named **dbd01** are renamed to **dbd02**:

```
DBD=(dbd01,dbd02)
```

IDMS_PAGEGROUP_RADIX=(old_page_group,new_page_group,old_radix,new_radix)

For IDMS data maps, changes the page group or radix value. This parameter enables you to migrate registrations and extraction maps to environments with different values for the page group and radix database settings.

For *old_page_group* and *new_page_group*, specify a value from 0 to 32767.

For *old_radix* and *new_radix*, specify a value are 2 to 12.

MAP=(old_map_name,new_map_name)

Renames the data maps that match the name or pattern in the first operand to the specified name.

In the following example, all data maps named **map01** are renamed to **map02**:

```
MAP=(map01,map02)
```

In the following example, all data maps are renamed to **newmap**:

```
MAP=(*,newmap)
```

In the following example, all data map names ending in **tmp** are renamed to **fixed**:

```
MAP=(*tmp,fixed)
```

SCHEMA=(old_schema_name,new_schema_name)

Renames the data map schemas that match the name or pattern in the first operand to the specified name.

In the following example, all schemas named **test** are renamed to **prod**:

```
SCHEMA=(test,prod)
```

In the following example, all schemas are renamed to **newprod**:

```
SCHEMA=(*,newprod)
```

In the following example, all schemas ending in **tmp** are renamed to **fixed**:

```
SCHEMA=(*tmp,fixed)
```

TABLE=(old_table_name,new_table_name)

Renames the tables that match the name or pattern in the first operand to the specified name.

In the following example, all tables named **testtab01** are renamed to **prodtab01**:

```
TABLE=(testtab01,prodtab01)
```

In the following example, all tables are renamed to **newtable**:

```
TABLE=(*,newtable)
```

In the following example, all tables ending in **01** are renamed to **fixed**:

```
TABLE=(*01,fixed)
```

DM_COPY SELECT Statement

The SELECT statement specifies filter criteria for the data maps to be copied.

The SELECT statement has the following parameters:

AM=access_method

Specifies the access method of the data map.

The following table lists and describes the values for *access_method*:

Access Method	Data Source
ADABAS	Adabas
DB2	DB2 for z/OS
DB2UDB	DB2 for Linux, UNIX, and Windows
DB2UNLD	DB2 Unload
DCOM	Datacom
DL1	DL/1 batch for IMS
ESDS	VSAM ESDS
IDMS	IDMS
IMS	IMS

Access Method	Data Source
KSDS	VSAM KSDS
MSSQL	Microsoft SQL Server
ODBA	IMS ODBA
Oracle	Oracle
RRDS	VSAM RRDS
SEQ	Sequential data set

MAP=map_name

Specifies a map name to select, which is any of the following:

- The full name of a data map
- A partial name with a wildcard (*)
- A wildcard (*) only

The following example specifies the map named **sample**:

```
MAP=sample
```

The following example specifies maps beginning with **sam**:

```
MAP=sam*
```

The following example specifies all data maps:

```
MAP=*
```

The default is *.

SCHEMA=schema_name

Specifies a schema name to select, which is one of the following:

- The full name of a data map schema
- A partial name with a wildcard (*)
- A wildcard (*) only

The following example specifies the schema **db2map**:

```
SCHEMA=db2map
```

The following example specifies schemas beginning with **prod**:

```
SCHEMA=prod*
```

The following example specifies all schemas:

```
SCHEMA=*
```

The default is *.

REG_COPY Statement

Use the REG_COPY statement to copy capture registrations. This statement has no operands but can be followed by other statements that filter the selected objects, rename objects, change object attributes, or enable optional functions.

The following table summarizes the optional statements that can follow the REG_COPY statement:

Optional Statement	Parameters
[CHECKXREF]	None
[CREATEXMAPS]	[LOC={SOURCE TARGET}] [OUTPUT= <i>alternative_pathname/data_set</i>]
[EXCLUDE]	[DBID= <i>database_instance</i>] [DBTYPE= <i>database_type</i>] [REG_NAME= <i>registration_name</i>]
[FASTLOAD]	None
[KEEPREGTAG]	None
[MODIFY]	[CONDENSE={FULL PART NONE}] [DBID= <i>database_ID</i>] [DBNAME= <i>database_name</i>] [FILEID= <i>file_ID</i>] [FN= <i>file_name</i>] [MSSOPTS=(DBSERVER= <i>db_server</i> ,DBNAME= <i>database_name</i>)] [MUFNAME= <i>muf_name</i>] [NEW_DBID= <i>database_instance</i>] [SUBSCHEMA= <i>subschema_name</i>] Note: In the MSSOPTS parameter, you can include a port number in the following format: DBSERVER=" <i>db_server,port_number</i> ",DBNAME= <i>database_name</i>
[RELATED]	BULK
[RENAME]	[BULKMAP=(<i>old_map_name,new_map_name</i>)] [BULKSCHEMA=(<i>old_schema_name,new_schema_name</i>)] [BULKTABLE=(<i>old_table_name,new_table_name</i>)] [DBD=(<i>old_dbd_name,new_dbd_name</i>)] [IDMS_PAGEGROUP_RADIX=(<i>old_page_group,new_page_group,old_radix,new_radix</i>)] [IMSMAP=(<i>old_map_name,new_map_name</i>)] [IMSSCHEMA=(<i>old_schema_name,new_schema_name</i>)] [NRDB_DM_TABLE=(<i>old_map_name,new_map_name,old_table_name,new_table_name</i>)] [REG_NAME= <i>old_registration_name,new_registration_name</i>] [SCHEMA=(<i>old_schema_name,new_schema_name</i>)] [TABLE=(<i>old_table_name,new_table_name</i>)]
[SELECT]	[DBID= <i>database_instance</i>] [DBTYPE= <i>database_type</i>] [REG_NAME= <i>registration_name</i>]

Notes:

- Do not use the OUTPUT statement with the REG_COPY statement if you also use the CREATEXMAPS statement.

- To use REG_COPY to copy registrations to an alternative CCT data set and create extraction maps, you must connect to the target by using a PowerExchange Listener, rather than using the default of local in the TARGET statement.
- If you include any of the following statements or combinations of statements and options after the REG_COPY statement, DTLURDMO reads the IMS data map from the target system:
 - RENAME IMSSHEMA
 - RENAME IMSMAP
 - CHECKXREF
- If you copy a registration that has been suspended with the PWXUCREG utility, the following processing occurs:
 - If the current registration status is Suspended, the status is reset to Active.
 - The activation and suspension timestamps that define the suspension window are cleared.
- If you use the FASTLOAD statement, the instance of the DTLURDMO utility that includes FASTLOAD must be the only operation that is updating the CCT file.

REG_COPY CHECKXREF Statement

For IMS data sources, the REG_COPY CHECKXREF statement forces the DTLURDMO utility to load the corresponding data map on the target system and to update the registrations with the database organization information from the DBD that is specified in the data map when copying registrations. Use the CHECKKXREF parameter if you change the DBD to specify a different database organization and then need to change the registrations.

This statement has no parameters.

Important: When you run the DTLURDMO utility with a REG_COPY CHECKXREF statement in the SYSIN and a TARGET value of LOCAL, ensure that the IMS RESLIB data set, or whichever library contains module DFSVC000 in your installation, is specified in the STEPLIB concatenation in the DTLURDMO JCL. If the TARGET value specifies a PowerExchange Listener node, ensure that the IMS RESLIB data set, or whichever library contains module DFSVC000, is specified in the STEPLIB concatenation in the target Listener JCL.

REG_COPY CREATEXMAPS Statement

The CREATEXMAPS statement creates an extraction map on the target system.

Use the following syntax:

```
CREATEXMAPS
[LOC={SOURCE|TARGET}]
[OUTPUT=alternative_pathname|data_set]
;
```

The REG_COPY statement has the following optional parameters:

LOC={SOURCE|TARGET}

Specifies whether the data map used to create the extraction map is loaded from the source or target location. Default is SOURCE.

OUTPUT=*alternative_pathname|data_set*

The extraction map is written to the alternative location. This function is analogous to that provided by the OUTPUT statement.

REG_COPY FASTLOAD Statement

Use the FASTLOAD statement with REG_COPY to improve performance when DTLURDMO is the only operation updating the CCT file.

When you include the FASTLOAD statement, the utility writes registration to the target CCT file at a consistent rate, regardless of the number of registrations. Without the FASTLOAD statement, utility processing runs slower when registrations are written to the CCT file, because the utility performs validations and prevents duplicate registrations for the same name or database resource. Informatica recommends using FASTLOAD if the current execution of the DTLURDMO utility is the only process updating registrations in the CCT file.

Use the following syntax:

```
FASTLOAD;
```

Important: If you use this statement, make sure that DTLURDMO is the only process updating the CCT file. If another process tries to update the CCT file concurrently with DTLURDMO, the FASTLOAD parameter can cause errors, or the utility might fail to detect duplicate registrations capturing data from the same database resource.

REG_COPY EXCLUDE Statement

The EXCLUDE statement specifies filter criteria for excluding registrations from the copy operation.

The statement is optional. By default, no items are excluded.

The statement has the following parameters:

DBID=database_instance

The database instance associated with the registration or registrations to be exclude. This value depends on the database type. For example, the database instance might be a DB2 subsystem ID or a database name. You can use the asterisk (*) wildcard character to select multiple instances.

DBTYPE=database_type

The database type associated with the registration or registrations to exclude.

Valid values are:

- **ADA** for Adabas.
- **AS4** for DB2 for i (i5/OS).
- **DB2** for DB2 for z/OS.
- **DCM** for Datacom.
- **IDM** for IDMS.
- **IMS** for IMS.
- **MSS** for Microsoft SQL Server.
- **MYS** for MySQL.
- **ORA** for Oracle.
- **PGS** for PostgreSQL.
- **UDB** for DB2 for Linux, UNIX, and Windows.
- **VSM** for VSAM.

REG_NAME=registration_name

Specifies the name of the registration to exclude. You can use the asterisk (*) wildcard character to select multiple registrations.

Note: The parameters for the EXCLUDE statement are the same as those for the SELECT statement.

RELATED TOPICS:

- [“REG_COPY SELECT Statement” on page 184](#)

REG_COPY KEEPREGTAG Statement

The KEEPREGTAG statement retains the original registration tag from the registration being copied when generating the extraction map. This statement is valid only with the CREATEXMAPS statement.

Do not use the KEEPREGTAG statement with the following REG_COPY statements:

- CHECKXREF
- MODIFY MUF_NAME
- MODIFY NEW_DBID
- RENAME DBD
- RENAME IMSMAP
- RENAME IMSSHEMA
- RENAME REG_NAME

These statements might cause a different registration tag to be generated for the copied registration and the extraction map.

REG_COPY MODIFY Statement

The MODIFY statement modifies certain attributes of the copied registrations.

This statement can include the following optional parameters:

CONDENSE={FULL|PART|NONE}

Specifies condense options for the captured data on the target system.

DBID=database_ID

For Adabas and Datacom databases, specifies a new database identifier.

DBNAME=database_name

For IDMS databases, specifies a new database name for the registration.

FN=file_name

Specifies the file name associated with the registration. For example:

```
FN=NEW.KSDS.FILE001
```

MSSOPTS=(DBSERVER=database_server,DBNAME=database_name)

Specifies the following Microsoft SQL Server options for selecting the registration to modify:

- DBSERVER specifies the database server ID for the registration. You can optionally include a port number in the following format:

```
DBSERVER="database_server,port_number"
```

In this case, the double-quotation marks are required.

- DBNAME specifies the database name for the registration.

MUFNAME=*muf_name*

For Datacom databases, specifies a new MUF name for the registration. Use this parameter to specify a new database instance for the registration in the same way you can use the NEW_DBID parameter for other database types.

NEW_DBID=*database_ID*

Specifies a new database instance for the registration. This value depends on the source type. Usually, it is the instance value from the registration group.

For Microsoft SQL Server sources, this value is a unique user-defined instance identifier for the database server and database name combination defined in the MSSOPTS parameter. Maximum length is seven characters. This instance identifier identifies a set of registrations for the publication database on the target system. If you also define the CREATEXMAPS statement in the registration copy syntax, the instance identifier is incorporated into the names of the extraction maps that are generated for the copied registrations on the target. If you use the PowerExchange Logger for Linux, UNIX, and Windows, ensure that the instance identifier matches the DBID parameter value in the Logger configuration file on the target. If you do not enter a NEW_DBID value, PowerExchange generates a unique instance identifier that is composed of all or part of the publication database name followed by a 3-digit number if a number is required to make the identifier unique.

Tip: If you are migrating from one SQL Server environment to another, you can enter an instance identifier that matches the instance identifier in the source system. In this manner, you can avoid using a generated instance identifier and having to update the extraction map names in PowerCenter workflows and edit the PowerExchange Logger DBID parameter value on the target. In this case, ensure that the DBMOVER configuration files in the source and target environments define unique paths in the CAP_PATH and CAPT_XTRA statements.

SUBSCHEMA=*subschema_name*

For IDMS databases, specifies a new subschema for the registration.

REG_COPY RELATED BULK Statement

The RELATED BULK statement merges the extraction map with an existing bulk data map on the target system.

This statement is valid only in conjunction with the CREATEXMAPS statement for DB2 for z/OS or DB2 for Linux, UNIX, and Windows registrations.

Use the following syntax:

```
RELATED BULK;
```

The name of the bulk data map that DTLURDMO looks for on the target system depends on whether the original extraction map has been merged with a data map on the source system:

- If the original extraction map associated with the registration on the source system was merged with a bulk data map, DTLURDMO uses the same data map and table name when merging the generated extraction map on the target system.
- If a bulk data map was not merged with the original extraction map on the source system, or an extraction map does not exist, a bulk data map name of the following form is generated for the merge:

table_name.registration_name_table_name

If DTLURDMO does not find the bulk data map on the target system, DTLURDMO reports the error and continues.

You can use the RENAME statement to modify the name of the generated extraction map or the name of the bulk data map to be merged with it.

REG_COPY RENAME Statement

The RENAME statement renames elements of the copied capture registration on the target system or identifies an existing bulk data map on the target system that is named differently from the default.

- For the DBD, IMSMAP, IMSSHEMA, REG_NAME, SCHEMA, and TABLE parameters, the RENAME statement specifies the new name of the registration element on the target system.
- For the BULKSCHEMA, BULKMAP, and BULKTABLE parameters, the RENAME statement identifies the bulk data map on the target system to be merged with the newly generated extraction map. Use these parameters if the bulk data map on the target system is named differently from the default.

These parameters are available only on DB2 for z/OS or DB2 for Linux, UNIX, and Windows systems.

For most RENAME parameters, the first operand represents the item or items being renamed, and the second operand represents the new name. The first operand can be any of the following:

- The full name of an item
- A partial name with a wildcard (*)
- A wildcard (*) only

The RENAME statement has the following parameters:

BULKMAP=(old_map_name,new_map_name)

Specifies a new map name to use in locating the bulk data map on the target system to merge with the copied extraction map. The BULKMAP parameter of the RENAME statement is valid only in conjunction with the RELATED BULK statement.

In the following example, all maps named **capture1** are renamed to **capture2**:

```
BULKMAP=(capture1,capture2)
```

In the following example, all maps are renamed to **newmap**:

```
BULKMAP=(*,newmap)
```

In the following example, all map names ending in **01** are renamed to fixed:

```
BULKMAP=(*01,fixed)
```

BULKSCHEMA=(old_schema_name,new_schema_name)

Specifies a new schema name to use in locating the bulk data map on the target system to merge with the copied extraction map. The BULKSCHEMA parameter of the RENAME statement is valid only in conjunction with the RELATED BULK statement.

In the following example, all bulk schemas named **test** are renamed to **prod**:

```
BULKSCHEMA=(test,prod)
```

In the following example, all bulk schemas are renamed to **newprod**:

```
BULKSCHEMA=(*,newprod)
```

In the following example, all schemas ending in **tmp** are renamed to **fixed**:

```
BULKSCHEMA=(*tmp,fixed)
```

BULKTABLE=(old_table_name,new_table_name)

Specifies a new table name to use in locating the bulk data map on the target system to merge with the copied extraction map. The BULKTABLE parameter of the RENAME statement is valid only in conjunction with the RELATED BULK statement.

In this example, all table names **testtab01** are renamed to **prodtab01**:

```
BULKTABLE=(testtab01t,prodtab01)
```

In this example, all tables are renamed to **newtable**:

```
BULKTABLE=(*,newtable)
```

In this example, all tables ending in **01** are renamed to **fixed**:

```
BULKTABLE=(*01,fixed)
```

DBD=(old_dbd_name,new_dbd_name)

For IMS data sources, specifies a new DBD name for the capture registration. DTLURDMO loads the corresponding DBD on the target system and updates the registration with the database organization.

The DBD specified in a RENAME statement takes precedence over the DBD name derived from a map specified by IMSSHEMA or IMSMAP.

IDMS_PAGEGROUP_RADIX=(old_page_group,new_page_group,old_radix,new_radix)

For IDMS data sources, changes the page group or radix value. This parameter enables you to migrate registrations and extraction maps to environments with different values for the page group and radix database settings.

For *old_page_group* and *new_page_group*, specify a value from 0 to 32767.

For *old_radix* and *new_radix*, specify a value from 2 to 12.

Note: You can create an extraction map on the target system with renamed page group and radix values by using the REG_COPY, CREATEXMAPS LOC=TARGET, and RENAME IDMS_PAGEGROUP_RADIX statements. Before you issue these statements, you must copy the data map to the target platform by using the DM_COPY statement. If the data map is not correct on the target platform, the results of the REG_COPY and accompanying optional statements are unpredictable.

IMSMAP=(old_map_name,new_map_name)

Renames IMS data maps that match the name or the pattern in the old data map name to the new data map name. The utility loads the specified data map from the target system and updates the registration with database organization using the DBD named in the data map.

In the following example, all schemas named **test** are renamed to **prod**:

```
IMSMAP=(test,prod)
```

IMSSHEMA=(old_schema_name,new_schema_name)

Renames IMS schemas that match the name or pattern in the old schema name to the new schema name. The utility loads the specified data map from the target system and uses the DBD named in the data map to update the database organization information in the registration.

The new schema name must match the name of the schema for an existing bulk data map on the target system.

In the following example, all schemas named **test** are renamed to **prod**:

```
IMSSHEMA=(test,prod)
```

NRDB_DM_TABLE=(old_map_name,new_map_name,old_table_name,new_table_name)

For IDMS data sources, identifies the old and new data map and table names. This parameter enables the registration to link to a data map that has a new identity in the new environment.

NRDB_DM_TABLE is required only if the data map contains information that is necessary for CDC extractions. Specifically, it is required to propagate changes to the page group or radix value to the extraction map.

The following rules apply:

- Use the following format for *old_map_name* and *new_map_name*:

schema_name.data_map_name

schema_name, *data_map_name*, or both can consist in part or full of wildcards, as in the following examples:

```
*.test_map
test_schema.*
test*,*
```

- If you omit *old_map_name* and *new_map_name*, include commas as placeholders, as follows:

NRDB_DM_TABLE=(, ,old_table_name,new_table_name)

- You can omit *old_table_name* and *new_table_name*. If you include them, specify the complete table name, without wildcards.

REG_NAME=(old_registration_name,new_registration_name)

Renames the registration. Because the registration name is used to create the registration tag, this option overrides the KEEPREGTAG option.

SCHEMA=old_schema_name,new_schema_name

Renames schemas that match the name or the pattern in the old schema name to the new schema name.

In the following example, all schemas named **test** are renamed to **prod**:

```
SCHEMA=(test,prod)
```

In the following example, all schemas are renamed to **newprod**:

```
SCHEMA=(*,newprod)
```

In the following example, all schemas ending in **tmp** are renamed to **fixed**:

```
SCHEMA=(*tmp,fixed)
```

TABLE=old_table_name,old_table_name

Renames tables that match the name or the pattern in the old table name to the new table name.

In the following example, all tables named **testtab01** are renamed to **prodtab01**:

```
TABLE=(testtab01,prodtab01)
```

In the following example, all tables are renamed to **newtable**:

```
TABLE=(*,newtable)
```

In the following example, all tables ending in **01** are renamed to **fixed**:

```
TABLE=(*01,fixed)
```

RELATED TOPICS:

- [“REG_COPY RELATED BULK Statement” on page 180](#)

REG_COPY SELECT Statement

The SELECT statement specifies filter criteria for selecting the registrations to be copied.

The following parameters are optional, but you must enter at least one to select registrations for copying:

DBID=database_instance

the source database instance associated with the registrations to select. This value depends on the database type. For example, the database instance might be a DB2 subsystem ID or a database name. You can use the asterisk (*) wildcard character to select multiple instances.

The following entries are valid:

- The full name of a specific instance. For example:
`DBID=sample`
- Part of the instance name with the asterisk wildcard (*) to create a mask that matches one or more instances. For example:
`DBID=sam*`
- The wildcard (*) only for all instances:
`DBID=*`

Default is the wildcard (*) only.

DBTYPE=database_type

Specifies the source database type associated with the registrations to select.

Valid values are:

- **ADA** for Adabas.
- **AS4** for DB2 for i (i5/OS).
- **DB2** for DB2 for z/OS.
- **DCM** for Datacom.
- **IDM** for IDMS.
- **IMS** for IMS.
- **MSS** for Microsoft SQL Server.
- **MYS** for MySQL.
- **ORA** for Oracle.
- **PGS** for PostgreSQL.
- **UDB** for DB2 for Linux, UNIX, and Windows.
- **VSM** for VSAM.

REG_NAME=registration_name

The name of the registration that you want to select for the copy operation. Use the registration name that was entered when the registration was created. You can use the asterisk (*) wildcard character to select multiple registrations.

The following entries are valid:

- The full registration name. For example:

```
REG_NAME=capture1
```

- Part of the registration name with the asterisk wildcard (*) to create a mask that matches one or more registrations. For example:

```
REG_NAME=prod*
```

- The wildcard (*) only to select all registrations. For example:

```
REG_NAME=*
```

Default is the wildcard (*) only.

XM_COPY Statement

The XM_COPY statement copies extraction maps from the source system to the target system. This statement has no operands but can be followed by other statements that filter the selected objects, rename objects, or change object attributes.

The XM_COPY statement enables you to copy extraction maps without copying capture registrations. Typically, when you copy a capture registration, you copy the extraction map at the same time by including the CREATEXMAPS statement after the REG_COPY statement.

The following table summarizes the optional statements that can follow the XM_COPY statement:

Optional Statement	Parameters
[EXCLUDE]	[AM=access_method] [MAP=map_name] [SCHEMA=schema_name]
[MODIFY]	AM=access_method [DB2INSTANCE=db2_instance] [DB2TABLE=db2_table_name] [DBD=dbd_name] [DBID=database_ID] [DBNAME=database_name] [DDSNODENAME=ddsnode_name] [DICTNAME=dictionary_name] [FILEID=file_ID] [FN=file_name] [IMSID=ims_ID] [MUFNAME=muf_name] [PCB=pcb_name] [PROGNAME=program_name] [PSB=psb_name] [SUBSCHEMA=subschema_name]
[RENAME]	[IDMS_PAGEGROUP_RADIX=(old_page_group,new_page_group,old_radix,new_radix)] [MAP=(old_map_name,new_map_name)] [NRDB_DM_TABLE=(old_map_name,new_map_name,old_table_name,new_table_name)] [REG_NAME=(old_registration_name,new_registration_name,new_version)] [REGTAG=(old_regtag,new_regtag)] [SCHEMA=(old_schema_name,new_schema_name)] [TABLE=(old_table_name,new_table_name)]
[SELECT]	[AM=access_method] [MAP=map_name] [SCHEMA=schema_name]

XM_COPY EXCLUDE Statement

The EXCLUDE statement specifies filter criteria for extraction maps to be explicitly excluded from the copying process.

The EXCLUDE statement is optional. By default, no items are excluded.

The EXCLUDE statement has the following parameters:

AM=access_method

Specifies the access method of the extraction maps to exclude.

MAP=map_name

Specifies a map name to exclude.

SCHEMA=schema_name

Specifies a schema name to exclude.

The parameters for the EXCLUDE statement are the same as those for the SELECT statement.

XM_COPY MODIFY Statement

The MODIFY statement modifies various attributes of the copied extraction maps.

The AM= parameter is required with the MODIFY statement. Depending on the value that you specify for the AM= parameter, you can include additional parameters.

The MODIFY statement has the following parameters:

AM=access_method

Required. Specifies the access method to use for modifying extraction maps.

The following table shows the access methods and the parameters that are available for each access method:

Access Method	Available Parameters
ADABAS	DBID, FILEID
DB2	DB2TABLE, DB2INSTANCE
DB2UDB	DB2TABLE, DB2INSTANCE
DCOM	DBID, DBNAME, MUFNAME
DL1	DBD, IMSID
ESDS	FN
IDMS	SUBSCHEMA, DBNAME, PROGNAME, DICTNAME, DDSNODENAME
IMS	None
KSDS	FN
MSSQL	None

Access Method	Available Parameters
MYSQL	None
ODBA	DBD, IMSID, PSB, PCB
Oracle	None
PGS	None
RRDS	FN
SEQ	FN

DB2TABLE=*schema.table_name*

Modifies the DB2 schema or table name for the table mapped by the extraction map. For example:

```
DB2TABLE=DSN8910.EMP
```

DB2INSTANCE=*instance*

Modifies the DB2 database name or subsystem ID for the data mapped by the extraction map. For example:

```
DB2INSTANCE=sample
```

DBD=*dbd_name*

Modifies the DBD name for the data mapped by the extraction map. For example:

```
DBD=PROD001
```

DBID=*database_ID*

Specifies a new database ID to use when reading the database.

DBNAME=*database_name*

Specifies a new database name to use when reading the database.

DDSNODENAME=*ddsnode_name*

Specifies a new DDSNODE name to use when reading the IDMS database.

DICTNAME=*dictionary_name*

Specifies a new dictionary name to use when reading the IDMS database.

FILEID=*file_ID*

Specifies a new ADABAS file ID for the data mapped by the extraction map.

FN=*filename*

Specifies a new name for the data file associated with the extraction map. The file name must be a valid file name on the target system. The following examples specify a new data file name for z/OS, VSAM, and Windows, respectively:

```
FN='DATA01.SEQ.FILE'
FN='SYS01.KSDS.DATA.FILE'
FN=c:\myfolder\myfile.txt
```

IMSID=*ims_ID*

Specifies a new IMS system ID for the data mapped by the extraction map. For example:

```
IMSID=IMS7
```

MUFNAME=*muf_name*

Specifies a new MUF name to use when reading the Datacom database.

PROGNAME=*program_name*

Specifies a new program name to use when reading the IDMS database.

PCB=*pcb_name*

Specifies a new PCB name for the data mapped by the extraction map. For example:

```
PCB=PCB020
```

PSB=*psb_name*

Specifies a new PSB for the data mapped by the extraction map. For example:

```
PSB=DTL003
```

SUBSCHEMA=*subschema_name*

Specifies a new subschema name to use when reading the IDMS database.

XM_COPY RENAME Statement

The RENAME statement specifies which elements of the extraction map name are renamed on the target system. For most parameters, the first operand represents the item or items being renamed, and the second operand represents the new name. For example:

```
MAP=(map01,map02)
```

The first operand can be any of the following:

- The full name of a schema, extraction map, or table
- A partial name with a wildcard (*)
- A wildcard (*) only

In each case, all items matching the specified name or pattern are renamed to the value of the second operand.

IDMS_PAGEGROUP_RADIX=(*old_page_group,new_page_group,old_radix,new_radix*)

For IDMS data sources, changes the page group or radix value. This parameter enables you to migrate registrations and extraction maps to environments with different values for the page group and radix database settings.

For *old_page_group* and *new_page_group*, specify a value from 0 to 32767.

For *old_radix* and *new_radix*, specify a value from 2 to 12.

Note: You can create an extraction map on the target system with renamed page group and radix values by using the XM_COPY and RENAME IDMS_PAGEGROUP_RADIX statements. Before you issue these statements, you must copy the data map to the target platform by using the DM_COPY statement. If the data map is not correct on the target platform, the results of the XM_COPY and RENAME IDMS_PAGEGROUP_RADIX statements are unpredictable.

MAP=(*old_map_name,new_map_name*)

Renames the extraction maps that match the name or pattern in the first operand to the specified name.

In the following example, all extraction maps named **map01** are renamed to **map02**:

```
MAP=(map01,map02)
```

In the following example, all extraction maps are renamed to **newmap**:

```
MAP=(*,newmap)
```

In the following example, all extraction map names ending in **tmp** are renamed to **fixed**:

```
MAP=(*tmp,fixed)
```

NRDB_DM_TABLE=(old_map_name,new_map_name,old_table_name,new_table_name)

For IDMS data sources, identifies the old and new data map and table names. This parameter enables the registration to link to a data map that has a new identity in the new environment.

NRDB_DM_TABLE is required only if the data map contains information that is necessary for CDC extractions. Specifically, it is required to propagate changes to the page group or radix value to the extraction map.

The following rules apply:

- Use the following format for *old_map_name* and *new_map_name*:

```
schema_name.data_map_name
```

schema_name, *data_map_name*, or both can consist in part or full of wildcards, as in the following examples:

```
*.test_map
test_schema.*
test*.*
```

- If you omit *old_map_name* and *new_map_name*, include commas as placeholders, as follows:

```
NRDB_DM_TABLE=(,,old_table_name,new_table_name)
```

- You can omit *old_table_name* and *new_table_name*. If you include them, specify the complete table name, without wildcards.

REG_NAME=(old_registration_name,new_registration_name,new_version)

Renames the registration and, optionally, the version. Because the registration name is used to create the registration tag, this option overrides the KEEPREGTAG option. This parameter provides a documented link to the new registration.

REGTAG=(old_regtag,new_regtag)

Renames the registration tags that match the name or pattern in the first operand to the specified name.

SCHEMA=(old_schema_name,new_schema_name)

Renames the schemas that match the name or pattern in the first operand to the specified name.

In the following example, all schemas named **test** are renamed to **prod**:

```
SCHEMA=(test,prod)
```

In the following example, all schemas are renamed to **newprod**:

```
SCHEMA=(*,newprod)
```

In the following example, all schemas ending in **tmp** are renamed to **fixed**:

```
SCHEMA=(*tmp,fixed)
```

TABLE=(old_table_name,old_table_name)

Renames the tables that match the name or pattern in the first operand to the specified name.

In the following example, all tables named **testtab01** are renamed to **prodtab01**:

```
TABLE=(testtab01,prodtab01)
```

In the following example, all tables are renamed to **newtable**:

```
TABLE=(*,newtable)
```

In the following example, all tables ending in **01** are renamed to **fixed**:

```
TABLE=(*01,fixed)
```

XM_COPY SELECT Statement

The SELECT statement specifies filter criteria for the data maps to be copied.

The statement has the following parameters:

AM=access_method

Specifies the access method of the extraction map.

The following table lists and describes the values for *access_method*:

Access Method	Data Source
ADABAS	Adabas
DB2	DB2 for z/OS
DB2UDB	DB2 for Linux, UNIX, and Windows
DCOM	Datacom
DL1	DL/1 batch for IMS
ESDS	VSAM ESDS
IDMS	IDMS
IMS	IMS
KSDS	VSAM KSDS
MSSQL	Microsoft SQL Server
MYSQL	MySQL
ODBA	IMS ODBA
Oracle	Oracle
PGS or PostgreSQL	PostgreSQL
RRDS	VSAM RRDS
SEQ	Sequential data set

MAP=map_name

Specifies an extraction map name.

The following entries are valid:

- The full name of a extraction map
- Part of an extraction map name with the asterisk wildcard (*) representing the remaining portion
- The asterisk wildcard (*) only

Default is the wildcard (*) only.

For example:

- The following entry specifies the extraction map named **sample**:

```
MAP=sample
```

- The following entry specifies extraction maps that have names beginning with **sam**:

```
MAP=sam*
```

- The following entry specifies all extraction maps:

```
MAP=*
```

SCHEMA=*schema_name*

Specifies a schema name to select, which is one of the following:

- The full name of a extraction map schema
- Part of the schema name with the wildcard (*) representing the remaining portion
- The asterisk wildcard (*) only

Default is the wildcard (*) only.

For example:

- The following entry specifies the schema **db2map**:

```
SCHEMA=db2map
```

- The following entry specifies schemas that have names beginning with **prod**:

```
SCHEMA=prod*
```

- The following entry specifies all schemas:

```
SCHEMA=*
```

Scope of Operands

Within the scope of a copy statement (DM_COPY, REG_COPY, or XM_COPY), DTLURDMO allows multiple occurrences of each of the following statements:

- EXCLUDE
- MODIFY
- RENAME
- SELECT

Multiple occurrences of these statements are supported to maintain backward compatibility and to provide flexibility when migrating a large number of objects. However, in most cases it is simpler and clearer to include multiple copy statements instead. This way, each copy statement is followed by at most one EXCLUDE, MODIFY, RENAME, and SELECT statement.

If you include multiple occurrences of the EXCLUDE MODIFY, RENAME, or SELECT statements, include a TESTMODE statement or a VALIDATE statement to run DTLURDMO in test mode and verify the operation of these statements. In addition, if you include multiple occurrences of the MODIFY or SELECT statements within the scope of a copy statement, PowerExchange issues a warning message to indicate that the outcome of these multiple statements might be unpredictable.

Behavior of EXCLUDE, MODIFY, RENAME, and SELECT Statements

The following rules summarize the behavior of the EXCLUDE, MODIFY, RENAME, and SELECT statements within the scope of a copy statement:

- To determine which objects to exclude, DTLURDMO performs a logical OR operation on all the EXCLUDE statements.
- To determine which objects to select, DTLURDMO performs a logical OR operation on all the SELECT statements.
- The EXCLUDE and SELECT statements determine the scope of each RENAME statement. Each RENAME statement renames the specified objects that are selected by the SELECT statements and not excluded by the EXCLUDE statements.
- The EXCLUDE and SELECT statements determine the scope of each MODIFY statement. The scope of each MODIFY statement is further restricted by the specified access method. Multiple MODIFY statements that specify different access methods are logically independent (OR operation).
- If multiple MODIFY statements specify the same access method, DTLURDMO ignores all but the first statement.
- Except in the case of multiple MODIFY statements that specify the same access method, the order of statements does not matter.

Example Using Multiple SELECT and MODIFY Statements

Suppose you need to copy several data maps that were defined with an access method of SEQ, and you need to modify the file name attribute in each data map.

The following statements might appear to be correct, but they do **not** produce the desired result:

```
DM_COPY;
SELECT MAP=fbti SCHEMA=vsam ;
MODIFY AM=KSDS FN=FPRSV.PAS.PSCODV1;
SELECT MAP=scpd SCHEMA= flatfile ;
MODIFY AM=SEQ FN=FPRS.PAS.D416.PSW.PLNXTTR;
SELECT MAP=sczp SCHEMA= flatfile ;
MODIFY AM=SEQ FN=FPRS.PAS.D416.PSW.PLNSTAT;
```

In this example, the scope of the MODIFY statements is determined by all the SELECT statements within the scope of the DM_COPY statement, not just the immediately preceding SELECT statement. The following statement is disregarded, because it applies to the same access method and set of selected data maps (as determined by the SELECT statements) as a previous MODIFY statement within the scope of a single DM_COPY command:

```
MODIFY AM=SEQ FN=FPRS.PAS.D416.PSW.PLNSTAT
```

Instead, include multiple DM_COPY commands, with each DM_COPY command followed by a single SELECT command and a single MODIFY command:

```
DM_COPY;
SELECT MAP=fbti SCHEMA=vsam ;
MODIFY AM=KSDS FN=FPRSV.PAS.PSCODV1;

DM_COPY;
SELECT MAP=scpd SCHEMA= flatfile ;
MODIFY AM=SEQ FN=FPRS.PAS.D416.PSW.PLNXTTR;

DM_COPY;
```



```
SELECT MAP=sczp SCHEMA= flatfile ;  
MODIFY AM=SEQ FN=FPRS.PAS.D416.PSW.PLNSTAT;
```

Running the DTLURDMO Utility on i5/OS

To run the DTLURDMO utility on i5/OS, enter the following command:

```
CALL PGM(DTLURDMO_executable_file_name)
```

For example:

```
CALL PGM(dtllib/DTLURDMO)
```

By default, the DTLURDMO utility looks for the DTLURDMO definition file in the CFG(DTLURDMO) member in the current *datalib* library. The DTLURDMO definition file contains the DTLURDMO control statements.

To specify an alternative location for the DTLURDMO definition file, enter the library name and file name of the definition file in the PARM option. For example:

```
CALL PGM(dtllib/DTLURDMO) PARM ('datalib/definition_file(DTLURDMO)')
```

Running the DTLURDMO Utility on Linux, UNIX, and Windows

On Linux, UNIX, or Windows, run the utility by navigating to the Informatica PowerExchange directory and entering `dtlurdm` on the command line as follows:

```
dtlurdm DTLURDMO_definition_file
```

For example:

```
dtlurdm e:\powerexchange\bin\dtlurdm.ini
```

The DTLURDMO definition file contains the DTLURDMO control statements. If no definition file is specified, PowerExchange looks for the `dtlurdm.ini` file in the current path.

Source and target locations specified in the control statements can include z/OS and IBM i systems, provided that user IDs and passwords associated submitted with the control statements have the appropriate authority to access maps and registrations on those systems. Performance is similar to running the utility on the remote system since network overhead for the utility is low.

Note: If you use the utility with the REGCOPY statement and do not include the FASTLOAD statement, utility performance can be slow. If you run the utility to copy registrations without the FASTLOAD statement, the system from which you launch the utility has no significant impact on performance. For information about the FASTLOAD statement, see [“REG_COPY FASTLOAD Statement” on page 178](#).

To monitor progress of the utility in a command window, use the REPORT_DEST STDOUT statement.

If you run DTLURDMO on Linux, UNIX, or Windows to process maps or registrations on remote z/OS or IBM i systems, you can also monitor the progress in the DTLLOG of the remote PowerExchange Listener. Message PWX-33304 indicates the start of a connection to the remote system, and messages PWX-00408 and PWX-00409 indicate the close of a result set.

Running the DTLURDMO Utility on z/OS

You run the utility by submitting the DTLURDMO job. The input control statements for this utility are read from SYSIN.

Note: To accommodate memory requirements for utility processing, specify a REGION size of 64M or larger.

The following is an example of JCL to use when you run this utility on z/OS.

```
//DTLUSR01 JOB 'ADA',MSGLEVEL=1,
//          CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
//*
//DTL JCLLIB ORDER=(DTLUSR.V951.RUNLIB)
//*
//          SET HLQ=DTLUSR.V951
//*
//URDMO    PROC HLQ=&HLQ
//*
//STEP1    EXEC PGM=DTLURDMO,
//          REGION=64M,TIME=NOLIMIT
//STEPLIB  DD DSN=CEE.SCEERUN,DISP=SHR
//          DD DSN=&HLQ..LOADLIB,DISP=SHR
//*DTLCAMAP DD DSN=&HLQ..DTLCAMAP,
//*          DISP=SHR
//*
//DTLMSG   DD DSN=&HLQ..DTLMSG,DISP=SHR
//DTLCFG   DD DSN=&HLQ..RUNLIB(DBMOVER),DISP=SHR
//DTLKEY   DD DSN=&HLQ..RUNLIB(LICENSE),DISP=SHR
//DTLSGN   DD DSN=&HLQ..RUNLIB(SIGNON),DISP=SHR
//DTLLOG   DD SYSOUT=*
//DATAMAP  DD DSN=&HLQ..V1.DATAMAPS,DISP=SHR
//DTLCAMAP DD DSN=&HLQ..V1.DTLCAMAP,DISP=SHR
//DTLREPOS DD DSN=&HLQ..V1.REPOS,DISP=SHR
//DTLAMCPR DD DSN=&HLQ..V1.CCT,DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//CEEDUMP  DD SYSOUT=*
//          PEND
//*
//*
//* EXECUTE THE PROCEDURE
//*
//DTLLSTN EXEC URDMO
//*
//SYSIN    DD *
REPORTDEST STDOUT;
USER DTLUSR;
EPWD 095E463AC1C5D5B8;
TARGET DTLUSR;
SOURCE NODE1;
OUTPUT DTLUSR.V951.V1.DATAMAPS.TESTMIGR;
DETAIL;
DM_COPY;
      SELECT AM=ADABAS;
//*
```

DTLURDMO Control Statement Examples

The following examples show the control statements for example DTLURDMO jobs.

Copying Selected Data Maps

The following example uses the DM_COPY statement to copy data maps from systema to systemb. The following conditions apply:

- Only data maps with the test01 schema and the DB2 access method are copied.
- The data map test01.map01 is excluded from the copy.

The schema name of the copied data map is changed from test01 to test04.

```
global statements
SOURCE systema;
TARGET systemb;
more global statements
DM_COPY;
SELECT SCHEMA=test01 AM=DB2;
EXCLUDE MAP=map01;
RENAME SCHEMA=(test01,test04);
```

Copying All Data Maps

The following example uses the DM_COPY statement to copy all data maps from systema to systemb.

```
global statements
SOURCE systema;
TARGET systemb;
more global statements
DM_COPY;
```

Copying and Modifying Data Maps

The following example uses DM_COPY to copy all data maps from systema to systemb. All data maps are modified to use the DSN6 subsystem ID.

```
global statements
SOURCE systema;
TARGET systemb;
more global statements
DM_COPY;
MODIFY AM=DB2 DB2INSTANCE=DSN6;
```

Copying Registrations and Generating Extraction Maps

The following example uses REG_COPY to copy registrations from systema to systemb and generate extraction maps on systemb. This example illustrates how to migrate registrations from a test system to a production system.

Because a SELECT statement is not included, all registrations are selected.

The schema name of the registered table is changed from test01 to prod01 on the target system, and the database instance is changed to DSNP.

```
global statements
SOURCE systema;
TARGET systemb;
more global statements
REG_COPY;
CREATEXMAPS;
RENAME SCHEMA=(test01,prod01);
MODIFY NEW_DBID=DSNP;
```

Copying Registrations, Generating Extraction Maps, and Merging Extraction Maps with Bulk Data Maps

The following example uses REG_COPY to copy a specific registration from systema to systemb and generate an extraction map on systemb.

Also, the RELATED BULK statement merges the created extraction map with a bulk data map on the target system. The RENAME statements identify the bulk data map on the target system.

```
global statements
SOURCE systema;
TARGET systemb;
more global statements
REG_COPY;
CREATEXMAPS;
RELATED BULK;
SELECT REG_NAME=capture01;
RENAME BULKSCHEMA=(*,test) BULKMAP=(*,map01) BULKTABLE=(*,table01);
```

These statements copy registration capture01, generate extraction map dbtestdb.capture01 and merge the extraction map with data map test.map01_table01. Because only one registration is selected, you can use wildcards in the RENAME statement to explicitly force DTLURDMO to merge with the required bulk map.

You can use subsequent input REG_COPY statements to repeat the process for other registrations.

Copying Microsoft SQL Server Registrations and Generating Extraction Maps with a User-Defined Instance ID

This example copies Microsoft SQL Server registrations from systema to the local target system and generates corresponding extraction maps on target. The extraction map names include the unique instance identifier that you define in the NEW_DBID parameter for the database server and database name combination.

The example uses the following syntax:

```
global statements
SOURCE systema;
TARGET local;
DETAIL;
REPLACE;
REG_COPY;
KEEPREGTAG;
CREATEXMAPS;
SELECT DBTYPE=MSS DBID=CAPT123;
MODIFY NEW_DBID=CAPTUR2,MSSOPTS=(DBSERVER=ABC188888\server2,DBNAME=capture2);
```

In this syntax:

- The REG_COPY statement copies the registrations.
- The CREATEXMAPS statement generates the extraction maps.
- The SELECT statement must contain DBTYPE=MSS to identify the source type as Microsoft SQL Server.
- The MODIFY statement contains the NEW_DBID and MSSOPTS parameters for this example:
 - The NEW_DBID parameter specifies the optional user-defined instance identifier for the database server and database name that are defined in the MSSOPTS parameter. This instance identifier has a maximum length of seven characters. You can enter only one unique instance identifier for a database server and database name combination. If you are migrating from one environment to another, you can enter an instance identifier that matches the instance identifier in the source system. In this manner, you can avoid using a generated instance identifier and having to update the extraction map names in PowerCenter workflows and edit the PowerExchange Logger DBID parameter value on the target.

- The MSSOPTS parameter identifies the SQL Server database server and database name.

Copying IMS Data Maps and Copying and Modifying Registrations

The following example runs DTLURDMO twice: first with a DM_COPY statement to copy data maps, and then with a REG_COPY statement to copy registrations and extraction maps.

The first execution of DTLURDMO uses the following input statements:

```
global statements
SOURCE systema;
TARGET systemb;
more global statements
DETAIL;
DM_COPY;
SELECT AM=DL1 SCHEMA=REGRESS MAP=FDPVF2;
MODIFY AM=DL1 PCBNUM=6;
RENAME SCHEMA=(REGRESS,IMSSRB);
```

These statements achieves the following results:

- The DM_COPY and SELECT statements copy the IMS DL/1 data map named REGRESS.FDPVF2 from the source system (PowerExchange Listener systsema) to the target system (PowerExchange Listener systemb).
- The RENAME SCHEMA statement changes the schema name from REGRESS to IMSSRB. The new data map name on the target is thus IMSSRB.FDPVF2.
- The MODIFY PCBNUM statement changes the PCB number in the data map on the target system to 6.

After DTLURDMO copies the data map from the source system to the target, a second exeuction of DTLURDMO copies the registration. This execution uses the following input statements:

```
global statements
SOURCE systema;
TARGET systemb;
more global statements
DETAIL;
REG_COPY;
CREATEXMAPS LOC=TARGET;
CHECKXREF;
SELECT REG NAME=DEPT DBID=SYNC DBTYPE=IMS;
RENAME IMSSCHEMA=(REGRESS,IMSSRB);
```

These statements achieve the following results:

- The REG_COPY and SELECT statements copy the IMS registration named DEPT that has the DBID and RECON ID of SYNC from the source system (PowerExchange Listener systema) to the target system (PowerExchange Listener systemb).
- The RENAME IMSSCHEMA statement changes the schema name from REGRESS to IMSSRB. The new data map name on the target is thus IMSSRB.FDPVF2.
- The CHECKXREF statement forces the utility to load the corresponding data map on the target system and to update the registration with database organisation from the DBD specified in the data map.
- CREATEXMAPS generates the new extraction map. This statement eliminate the need to run the utility again with the XM_COPY statement.

LOC=TARGET specifies that the data map used to create the extraction map is loaded from the target destination.

Copying IMS Data Maps and Registrations and Modifying the IMSID Data Map Property

The following example runs DTLURDMO twice: first with a DM_COPY statement to copy data maps, and then with a REG_COPY statement to copy registrations and extraction maps. The REG_COPY statement also modifies the IMSID data map property.

The DM_COPY and REG_COPY statements also modify the schema name and IMSID data map property. In this example, the source system uses IM95 for the schema name for IMS data maps and IM95 for the RECON ID on registration groups and for the IMS system ID. The target system uses IM91 for the schema name for IMS data maps and IM91 for the RECON ID on registration groups and for the IMS system ID.

The first execution of DTLURDMO uses the following input statements:

```
global statements
SOURCE systema;
TARGET systemb;
more global statements
DETAIL;
DM_COPY;
SELECT AM=DL1 SCHEMA=IM95;
MODIFY AM=DL1 IMSID=IM91;
RENAME SCHEMA=(IM95,IM91);
```

These statements achieve the following results:

- The DM_COPY and SELECT statements copy IMS data maps with schema name IM95 and access method DL1 from the source system (PowerExchange Listener systema) to the target system (PowerExchange Listener systemb).
- The RENAME SCHEMA statement changes the schema name from IM95 to IM91.
- The MODIFY statement modifies the IMSID data map property for all selected data maps for which AM=DL1 from IM95 to IM91.

After DTLURDMO copies the data map from the source system to the target, a second execution of DTLURDMO copies the registration. This execution uses the following input statements:

```
global statements
SOURCE systema;
TARGET systemb;
more global statements
DETAIL;
REG_COPY;
CREATEXMAPS LOC=TARGET;
SELECT DBID=IM95 DBTYPE=IMS;
MODIFY NEW_DBID=IM91;
RENAME IMSSCHEMA=(IM95,IM91);
```

These statements achieve the following results:

- The REG_COPY and SELECT statements copy IMS registrations with DBID of IM95 from the source system (PowerExchange Listener systema) to the target system (PowerExchange Listener systemb).
- CREATEXMAPS generates the new extraction map. This statement eliminates the need to run the utility with XM_COPY statement.

LOC=TARGET specifies that the data map used to create the extraction map is loaded from the target destination.
- The RENAME IMSSCHEMA statement changes the schema name from IM95 to IM91.
- The MODIFY NEW_DBID statement changes the IMSID data map property to IM91.

Note: When executing a REG_COPY statement, the DTLURDMO utility must load the IMS DBD if any of the following statements are also included:

- CHECKXREF
- RENAME DBD
- RENAME IMSMAP
- RENAME IMSSCHEMA

DTLURDMO retrieves the location of the IMS DBD library from the appropriate IMSID statement in the DBMOVER configuration file on the target system. The IMSID statement has the following syntax:

```
IMSID=(ims_ssid
      ,dbdlib
      [,RECON=(recon1
                [,recon2]
                [,recon3]))
)
```

DTLURDMO uses the IMSID statement that has a value for IMS_SSID that matches one of the following values, listed in order of priority:

1. If a MODIFY NEW_DBID statement is present, DTLURDMO uses this value, which represents the new data map property value.
2. If no MODIFY NEW_DBID statement is present, DTLURDMO uses the IMSID data map property. This value can be different from the RECON ID value.
3. If no MODIFY NEW_DBID statement is present, and the data map does not have a value for the IMSID property, DTLURDMO uses the value specified in the SELECT DBID= statement. This value is the value of the RECON ID of the registration group.
4. If none of the previous conditions apply, DTLURDMO issues a message indicating that the registration cannot be copied and will be skipped.

DTLURDMO Report Examples

The following examples show the report output format for the DTLURDMO utility when it runs in test mode, validation mode, and update mode.

DM_COPY Test Mode Example

The following example shows the test mode report for a DTLURDMO DM_COPY statement.

```
DTLURDMO DM_COPY    .\UpdateToLocal_ADA_T.ini
=====
Source: Location SYSBUSRA1      User USRA1      EPWD=****
Target: Location LOCAL         User USRA1      EPWD=****
Output  I:\datamaps_SYSBUSRA1

Detail      YES (Print a detailed report)
Testmode    YES (List maps/registrations but do not process them)
Validate    NO
Replace     YES

Number of DM_COPY items 2

Data map copy started

=====
```

```

DM_COPY # 1
  Exclude: Schema 'cpn301' Map 'SYSBUSRA1'
  Select:  Schema 'ada200' Access Method 'ADABAS'
  Rename:  Schema 'ada200' to 'XXXada200'
  Rename:  Table 'ADA_RECORD' to 'XXX_ADA_RECORD'
  Modify:  AM=ADA DBID=9200
  Modify:  AM=ADA FILENO=9001
=====

```

DMY selection SQL 'DBLIST DMX,ada200,*,SYSBUSRA1'

```

Data map 'ada200.ada200f001'  skipped because of TESTMODE
Data map 'ada200.save'        skipped because of TESTMODE

```

Totals for DM_COPY #1

```

=====
Data maps Excluded                0
Data maps Not Selected by name    0
Data maps Not Selected by access method 0
Data maps Selected                2
-----
Data maps Read                    2

Data maps Written                 0

```

DM_COPY # 2

```

  Exclude: Schema 'cpn301' Map 'SYSBUSRA1'
  Exclude: Schema 'ada200' Access Method 'ADABAS'
  Select:  Access Method 'ADABAS'
  Rename:  Table 'ADA_RECORD' to 'YYY_ADA_RECORD'
=====

```

DMY selection SQL 'DBLIST DMX,*,*,SYSBUSRA1'

```

Data map 'aaada1000.fdt50'      skipped because of TESTMODE
Data map 'ada200.ada200f001'    excluded by mask Schema 'ada200' Map ''
Data map 'ada200.save'          excluded by mask Schema 'ada200' Map ''
Data map 'bigims32.map1'        not selected because access method 'D' not in AM select list
Data map 'cleanse.charasc1'     not selected because access method 'S' not in AM select list
Data map 'cpn301.SYSBUSRA1'     excluded by mask Schema 'cpn301' Map 'SYSBUSRA1'
Data map 'cr384527.map2'        not selected because access method 'S' not in AM select list
Data map 'datamaps.hex1'        not selected because access method 'S' not in AM select list
Data map 'efld.mv001'           not selected because access method 'S' not in AM select list
Data map 'efld.mv002'           not selected because access method 'S' not in AM select list
...
Data map 'empss01.map1'         not selected because access method 'I' not in AM select list
Data map 'idmsqa.stcrss01'      not selected because access method 'I' not in AM select list
Data map 'imsesds.hspv16s2'     not selected because access method 'E' not in AM select list
Data map 'imskds.hspv16s2'      not selected because access method 'K' not in AM select list
Data map 'imskds.temp'          not selected because access method 'K' not in AM select list
Data map 'imsrds.hspv16s1'      not selected because access method 'N' not in AM select list
Data map 'imsrds.hspv16s2'      not selected because access method 'N' not in AM select list
Data map 'imsseq.dmnndout1'     not selected because access method 'S' not in AM select list
Data map 'imsseq.hspv16s1'      not selected because access method 'S' not in AM select list
Data map 'imsseq.hspv16s2'      not selected because access method 'S' not in AM select list
Data map 'imstape.hspv16s2'     not selected because access method 'T' not in AM select list
Data map 'ims9.dtl002'          not selected because access method 'O' not in AM select list
Data map 'ims9.dtl003'          not selected because access method 'O' not in AM select list
...
Data map 'ksds.general'         not selected because access method 'K' not in AM select list
...
Data map 'testnum.zonedmvs'     not selected because access method 'S' not in AM select list
Data map 'testout.zoned'        not selected because access method 'S' not in AM select list

```

Totals for DM_COPY #2

```

=====
Data maps Excluded                3
Data maps Not Selected by name    0
Data maps Not Selected by access method 217
Data maps Selected                1

```



```

-----
Data maps Read                                     221

Data maps Written                                 0

DTLURDMO DM_COPY Processing Totals
=====
Total data maps Excluded                         3
Total data maps Not Selected by name              0
Total data maps Not Selected by access method     217
Total data maps Selected                         3
-----
Total data maps Read                             223

Total data maps Written                           0

No map / registration components changed

```

DM_COPY Validate Example

The following example shows the validate mode report for a DTLURDMO DM_COPY statement:

```

DTLURDMO DM_COPY  .\UpdateToLocal_ADA_V.ini
=====
Source: Location SYSBUSRA1          User USRA1          EPWD=****
Target: Location LOCAL              User USRA1          EPWD=****
Output  I:\datamaps_SYSBUSRA1

```

```

Detail      YES (Print a detailed report)
Testmode    NO
Validate    YES (Process maps/registrations but do not save them)
Replace     YES

```

Number of DM_COPY items 2

Data map copy started

```

=====
DM_COPY # 1
  Exclude: Schema 'cpn301' Map 'sysbusra1'
  Select:  Schema 'ada200' Access Method 'ADABAS'
  Rename:  Schema 'ada200' to 'XXXada200'
  Rename:  Table 'ADA_RECORD' to 'XXX_ADA_RECORD'
  Modify:  AM=ADA DBID=9200
  Modify:  AM=ADA FILENO=9001
=====

```

DMY selection SQL 'DBLIST DMX,ada200,*,SYSBUSRA1'

1: Data map: Schema 'ada200' Map 'ada200f001' Access method 'ADABAS'

```

=====
SCHEMA ada200 changed to XXXada200
TABLE ADA_RECORD changed to XXX_ADA_RECORD
DBID 200 changed to 9200
FILENO 1 changed to 9001

```

Updated and renamed data map: Schema 'ada200' Map 'ada200f001' Changes 4
Tables: XXX_ADA_RECORD
DBID = 9200, FILENO = 9001
Save of map skipped because of VALIDATE=Y

2: Data map: Schema 'ada200' Map 'save' Access method 'ADABAS'

```

=====
SCHEMA ada200 changed to XXXada200
TABLE ADA_RECORD changed to XXX_ADA_RECORD
DBID 200 changed to 9200
FILENO 1 changed to 9001

```

Updated and renamed data map: Schema 'ada200' Map 'save' Changes 4

Tables: XXX_ADA_RECORD
 DBID = 9200, FILENO = 9001
 Save of map skipped because of VALIDATE=Y

Totals for DM_COPY #1
 =====

Data maps Excluded	0
Data maps Not Selected by name	0
Data maps Not Selected by access method	0
Data maps Selected	2

Data maps Read	2
Data maps Updated with changes	2
Data maps Written	0

=====

DM_COPY # 2
 Exclude: Schema 'cpn301' Map 'sysbusra1'
 Exclude: Schema 'ada200' Access Method 'ADABAS'
 Select: Access Method 'ADABAS'
 Rename: Table 'ADA_RECORD' to 'YYY_ADA_RECORD'
 =====

DMY selection SQL 'DBLIST DMX,*,*,SYSBUSRA1'

Data map 'a.notfound' not selected because access method 'S' not in AM select list

1: Data map: Schema 'aaada1000' Map 'fdt50' Access method 'ADABAS'
 =====

Copied data map: Schema 'aaada1000' Map 'fdt50'
 Tables: YYY_ADA_RECORD
 DBID = 1000, FILENO = 50
 Save of map skipped because of VALIDATE=Y

Data map 'ada200.ada200f001' excluded by mask Schema 'ada200' Map ''
 ...

Data map 'testout.zoned' not selected because access method 'S' not in AM select list

2: Data map: Schema 'xxxada200' Map 'ada200f001' Access method 'ADABAS'
 =====

Copied data map: Schema 'xxxada200' Map 'ada200f001'
 Tables: XXX_ADA_RECORD
 DBID = 9200, FILENO = 9001
 Save of map skipped because of VALIDATE=Y

3: Data map: Schema 'xxxada200' Map 'save' Access method 'ADABAS'
 =====

Copied data map: Schema 'xxxada200' Map 'save'
 Tables: XXX_ADA_RECORD
 DBID = 9200, FILENO = 9001
 Save of map skipped because of VALIDATE=Y

Totals for DM_COPY #2
 =====

Data maps Excluded	3
Data maps Not Selected by name	0
Data maps Not Selected by access method	218
Data maps Selected	3

Data maps Read	224
Data maps Copied without changes	3
Data maps Written	0

DTLURDMO DM_COPY Processing Totals
 =====

```

Total data maps Excluded                      3
Total data maps Not Selected by name          0
Total data maps Not Selected by access method 218
Total data maps Selected                      5
-----
Total data maps Read                          226

Total data maps Copied without changes        3
Total data maps Updated with changes          2
Total data maps Written                       0

Totals for map / registration components changed
=====
ADABAS DBID                                  2
ADABAS File number                          2
Schema                                       2
Table                                        2
-----
Total components changed                      8
DTLURDMO DM_COPY .\UpdateToLocal_ADA_V.ini
=====
Source: Location SYSBUSRA1          User USRA1          EPWD=****
Target: Location LOCAL              User USRA1          EPWD=****
Output  I:\datamaps_SYSBUSRA1

Detail      YES (Print a detailed report)
Testmode    NO
Validate    YES (Process maps/registrations but do not save them)
Replace     YES

Number of DM_COPY items 2

Data map copy started

=====
DM_COPY # 1
  Exclude: Schema 'cpn301' Map 'sysbusra1'
  Select:  Schema 'ada200' Access Method 'ADABAS'
  Rename:  Schema 'ada200' to 'XXXada200'
  Rename:  Table 'ADA_RECORD' to 'XXX_ADA_RECORD'
  Modify:  AM=ADA DBID=9200
  Modify:  AM=ADA FILENO=9001
=====

DMY selection SQL 'DBLIST DMX,ada200,*,sysbusra1'

1: Data map: Schema 'ada200' Map 'ada200f001' Access method 'ADABAS'
=====
SCHEMA ada200 changed to XXXada200
TABLE ADA_RECORD changed to XXX_ADA_RECORD
DBID 200 changed to 9200
FILENO 1 changed to 9001

Updated and renamed data map: Schema 'ada200' Map 'ada200f001' Changes 4
Tables: XXX_ADA_RECORD
DBID = 9200, FILENO = 9001
Save of map skipped because of VALIDATE=Y

2: Data map: Schema 'ada200' Map 'save' Access method 'ADABAS'
=====
SCHEMA ada200 changed to XXXada200
TABLE ADA_RECORD changed to XXX_ADA_RECORD
DBID 200 changed to 9200
FILENO 1 changed to 9001

Updated and renamed data map: Schema 'ada200' Map 'save' Changes 4
Tables: XXX_ADA_RECORD
DBID = 9200, FILENO = 9001
Save of map skipped because of VALIDATE=Y

Totals for DM_COPY #1

```

```

=====
Data maps Excluded                                0
Data maps Not Selected by name                    0
Data maps Not Selected by access method           0
Data maps Selected                                2
-----
Data maps Read                                    2

Data maps Updated with changes                    2
Data maps Written                                0

=====
DM_COPY # 2
  Exclude: Schema 'cpn301' Map 'sysbusra1'
  Exclude: Schema 'ada200' Access Method 'ADABAS'
  Select: Access Method 'ADABAS'
  Rename: Table 'ADA_RECORD' to 'YYY_ADA_RECORD'
=====

DMY selection SQL 'DBLIST DMX,*,*,SYSBUSRA1'

Data map 'a.notfound'          not selected because access method 'S' not in AM select list

1: Data map: Schema 'aaada1000' Map 'fdt50' Access method 'ADABAS'
=====

Copied data map: Schema 'aaada1000' Map 'fdt50'
Tables: YYY_ADA_RECORD
DBID = 1000, FILENO = 50
Save of map skipped because of VALIDATE=Y

Data map 'ada200.ada200f001'   excluded by mask Schema 'ada200' Map ''
...

Data map 'testout.zoned'       not selected because access method 'S' not in AM select list

2: Data map: Schema 'xxxada200' Map 'ada200f001' Access method 'ADABAS'
=====

Copied data map: Schema 'xxxada200' Map 'ada200f001'
Tables: XXX_ADA_RECORD
DBID = 9200, FILENO = 9001
Save of map skipped because of VALIDATE=Y

3: Data map: Schema 'xxxada200' Map 'save' Access method 'ADABAS'
=====

Copied data map: Schema 'xxxada200' Map 'save'
Tables: XXX_ADA_RECORD
DBID = 9200, FILENO = 9001
Save of map skipped because of VALIDATE=Y

Totals for DM_COPY #2
=====
Data maps Excluded                                3
Data maps Not Selected by name                    0
Data maps Not Selected by access method           218
Data maps Selected                                3
-----
Data maps Read                                    224

Data maps Copied without changes                    3
Data maps Written                                0

DTLURDMO DM_COPY Processing Totals
=====
Total data maps Excluded                          3
Total data maps Not Selected by name                0
Total data maps Not Selected by access method       218
Total data maps Selected                            5
-----

```

Total data maps Read	226
Total data maps Copied without changes	3
Total data maps Updated with changes	2
Total data maps Written	0
Totals for map / registration components changed	
=====	
ADABAS DBID	2
ADABAS File number	2
Schema	2
Table	2
-----	----
Total components changed	8

REG_COPY Test Mode Example

The following example shows the test mode report for a DTLURDMO REG_COPY statement:

```
DTLURDMO REG_COPY  .\UpdateToLocal_ADA_T.ini
=====
Source: Location SYSBUSRA1      User USRA1      EPWD=****
Target: Location LOCAL        User USRA1      EPWD=****
Output  I:\regcopy_sysbusral\regcopy_sysbusral (File I:
\regcopy_sysbusral\regcopy_sysbusralCCT.idx)

Detail      YES (Print a detailed report)
Testmode    YES (List maps/registrations but do not process them)
Validate    NO
Replace     NO
Fastload    NO

Number of REG_COPY items 1

=====
REG_Copy #1
Keepregtag      NO
Createxmaps     NO Xmap location ''
CheckXref       NO
Related         NO
      Select:  Database Type 'ADA' Database Instance '*' Reg name '*'
=====

CRAM_Get_Registration_List() location='SYSBUSRA1' dbtype='ADA' instance='*' crname='*'
version=0 status='*'
4 registrations returned

Registration 'ADA.ADA8242.adatemp'      selected by mask DB type 'ADA' instance '*'
crname '*'
Registration 'ADA.ADA8242.ada8242'      selected by mask DB type 'ADA' instance '*'
crname '*'
Registration 'ADA.ADA200.rvf0001'        selected by mask DB type 'ADA' instance '*'
crname '*'
Registration 'ADA.ADA200.rvf0002'        selected by mask DB type 'ADA' instance '*'
crname '*'

Totals for REG_COPY #1
=====
Registrations Excluded                      0
Registrations Not Selected by name          0
Registrations Not Selected by access method 0
Registrations Selected                      4
-----
Registrations Read                          4

Registrations Written                       0

DTLURDMO REG_COPY Processing Totals
```

```

=====
Total registrations Excluded                                0
Total registrations Not Selected by name                    0
Total registrations Not Selected by access method           0
Total registrations Selected                                4
-----
Total registrations Read                                    4

Total registrations Written                                  0

No map / registration components changed

```

REG_COPY Validate Mode Example

The following example shows the validate mode report for a DTLURDMO REG_COPY statement:

```

DTLURDMO REG_COPY    .\UpdateToLocal_ADA_V.ini
=====
Source: Location SYSBUSRA1      User USRA1      EPWD=****
Target: Location LOCAL         User USRA1      EPWD=****
Output  I:\regcopy_sysbusra1\regcopy_sysbusra1 (File I:
\regcopy_sysbusra1\regcopy_sysbusra1CCT.idx)

Detail      YES (Print a detailed report)
Testmode    NO
Validate    YES (Process maps/registrations but do not save them)
Replace     NO
Fastload    NO

Number of REG_COPY items 1

=====
REG_Copy #1
Keepregtag      NO
Createxmaps     NO Xmap location ''
CheckXref       NO
Related         NO
  Select: Database Type 'ADA' Database Instance '*' Reg name '*'
  Rename: Schema 'ada8242' to 'ada8242999'
  Modify: AM= NEW_DBID='9999'
=====

CRAM_Get_Registration_List() location='SYSBUSRA1' dbtype='ADA' instance='*' crname='*'
version=0 status='*'
4 registrations returned

Registration 'ADA.ADA8242.adatemp'      selected by mask DB type 'ADA' instance '*'
crname '*'

1: Registration: DB type 'ADA' Instance 'ADA8242' CR Name 'adatemp'
=====
Instance ADA8242 changed to 9999
Adabas File Nbr 1 changed to 888

Updated registration: DB type 'ADA' Instance '9999' CR Name 'adatemp' Changes 2
Save of registration skipped because of VALIDATE=Y
Registration 'ADA.ADA8242.ada8242'      selected by mask DB type 'ADA' instance '*'
crname '*'

2: Registration: DB type 'ADA' Instance 'ADA8242' CR Name 'ada8242'
=====
SCHEMA ada8242 changed to ada8242999
TABLE map351a_ADA_RECORD changed to map888_ADA_RECORDXXX
Instance ADA8242 changed to 9999
Adabas File Nbr 351 changed to 888

Updated registration: DB type 'ADA' Instance '9999' CR Name 'ada8242' Changes 4
Save of registration skipped because of VALIDATE=Y

```

```

Registration 'ADA.ADA200.rvf0001'      selected by mask DB type 'ADA' instance '*'
cname '*'

3: Registration: DB type 'ADA' Instance 'ADA200' CR Name 'rvf0001'
=====
Instance ADA200 changed to 9999
Adabas File Nbr 1 changed to 888

Updated registration: DB type 'ADA' Instance '9999' CR Name 'rvf0001' Changes 2
Save of registration skipped because of VALIDATE=Y
Registration 'ADA.ADA200.rvf0002'      selected by mask DB type 'ADA' instance '*'
cname '*'

4: Registration: DB type 'ADA' Instance 'ADA200' CR Name 'rvf0002'
=====
Instance ADA200 changed to 9999
Adabas File Nbr 1 changed to 888

Updated registration: DB type 'ADA' Instance '9999' CR Name 'rvf0002' Changes 2
Save of registration skipped because of VALIDATE=Y

Totals for REG_COPY #1
=====
Registrations Excluded                      0
Registrations Not Selected by name          0
Registrations Not Selected by access method 0
Registrations Selected                      4
-----
Registrations Read                          4

Registrations Updated with changes          4
Registrations Written                       0

DTLURDMO REG_COPY Processing Totals
=====
Total registrations Excluded                0
Total registrations Not Selected by name    0
Total registrations Not Selected by access method 0
Total registrations Selected                4
-----
Total registrations Read                    4

Total registrations Updated with changes    4
Total registrations Written                 0

Totals for map / registration components changed
=====
ADABAS File number                         4
Schema                                    1
Table                                    1
Instance                                  4
-----
Total components changed                    10

```

CHAPTER 14

DTLUTSK - Task Control Utility

This chapter includes the following topics:

- [DTLUTSK Utility Overview, 208](#)
- [DTLUTSK Command Line Utility on i5/OS, 208](#)
- [DTLUTSK Command Line Utility on Linux, UNIX, and Windows, 210](#)
- [DTLUTSK Job on z/OS, 212](#)
- [DTLUTSK Command Line Utility on z/OS, 214](#)
- [Running the DTLUTSK Utility in the PowerExchange Navigator, 216](#)
- [DTLUTSK Utility Security, 217](#)

DTLUTSK Utility Overview

Use this utility to list active tasks, current locations, or allocated data sets. You can also use this utility to stop active tasks for PowerExchange applications that read data for remote requests that run in the PowerExchange Listener.

You can run the DTLUTSK utility on any platform that PowerExchange supports by using the following methods:

- i5/OS command line
- Linux, UNIX, or Windows command line
- z/OS JCL job
- z/OS command line
- PowerExchange Navigator database row test

Note: To run the utility LISTTASK, STOPTASK, or LISTLOCATIONS command from the **Database Row Test** dialog box, select **TASK_CNTL** from the **DB_Type** list.

DTLUTSK Command Line Utility on i5/OS

Use the following syntax and parameters to run the DTLUTSK utility on i5/OS:

Syntax:

```
CALL PGM(library/DTLUTSK) PARM
('LOC=location
CMD=command_name
[ TASKID=task_id]
[ APPL=task_name]
[ NODETYPE={N|A|S}
[ UID=<user_id>]
[ PWD=<pwd_or_passphrase>]')
```

Tip: Be sure to add the *dtlib* and *datalib* libraries to your library list before executing a DTLUTSK command.

Parameters:

The following table describes the parameters:

Parameter	Description
LOC	The remote location where the task is running. Locations must be specified in NODE statements in the DBMOVER configuration file. If you enter LOCAL, the utility returns an error message.
CMD	The command name: <ul style="list-style-type: none">- LISTTASK. Lists all current tasks.- STOPTASK. Stops the task specified by TASKID parameter.- LISTLOCATIONS. Lists all current locations.- LISTALLOC. Lists all allocated data sets.
TASKID	When CMD=STOPTASK, the task ID of the task that you want to stop. You can determine the task ID by using the LISTTASK command.
APPL	When CMD=STOPTASK, the task name of the task that you want to stop.
NODETYPE	When CMD=LISTLOCATIONS, specify one of the following node types: <ul style="list-style-type: none">- N. Lists locations that are defined in NODE statements in the DBMOVER configuration file.- A. Lists locations that are defined in NODE or SVCNODE statements in the DBMOVER configuration file.- S. Lists locations that are defined in SVCNODE statements in the DBMOVER configuration file. Default is N.

Parameter	Description
UID	A user ID that has the authority to access the location, if required by your security settings.
PWD	<p>A password or encrypted password for the specified user. If a password contains nonalphanumeric characters, you must enclose it in double-quotation marks ("). Do not include double-quotation marks within a password string, even if you enclose it in double-quotation marks.</p> <p>For access to a remote i5/OS or z/OS location, you can enter a valid PowerExchange passphrase instead of a password. An i5/OS passphrase can be from 9 to 31 characters in length. A z/OS passphrase can be from 9 to 128 characters in length. Passphrases can contain the following characters:</p> <ul style="list-style-type: none"> - Uppercase and lowercase letters - The numbers 0 to 9 - Spaces - The following special characters: ' - ; # \ , . / ! % & * () _ + { } : @ < > ? <p>Note: The first character is an apostrophe.</p> <p>Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.</p> <p>If a passphrase contains spaces, enclose it in double quotation marks ("). If a passphrase contains special characters, enclose it in triple double-quotation marks (""").</p> <p>Note: On z/OS, a valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.</p> <p>To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the <i>PowerExchange Reference Manual</i>.</p>

DTLUTSK Command Line Utility on Linux, UNIX, and Windows

Use the following syntax and parameters to run the DTLUTSK utility on a Linux, UNIX, or Windows system:

Syntax:

```
DTLUTSK
  CMD=command_name
  [TASKID=task_id]
  [APPL=task_name]
  [NODETYPE={N|A|S}]
  LOC=location
  [UID=user_id]
  [PWD=password_or_passphrase]
```

Parameters:

The following table describes the parameters:

Parameter	Description
CMD	<ul style="list-style-type: none"> - LISTTASK. Lists all current tasks. - STOPTASK. Stops the task specified by TASKID parameter. - LISTLOCATIONS. Lists all current locations. - LISTALLOC. Lists the data sets allocated to a PowerExchange Listener on z/OS. Specify the remote z/OS node in the LOC parameter. If you attempt to run the command against a Listener on i5/OS, Linux, Unix, or Windows, the result set will be empty. <p>Alternatively, instead of issuing the LISTALLOC command from the command line, you can specify the LISTALLOC command in the Pre SQL or Post SQL attribute for a PowerCenter session to report file allocations when the workflow runs. Also specify the PowerCenter Retrieve PWX Log Entries connection attribute to enable the informational messages with file-allocation information to be written the session log.</p>
TASKID	When CMD=STOPTASK, the task ID of the task that you want to stop. You can determine the task ID by using the LISTTASK command.
APPL	When CMD=STOPTASK, the task name of the task that you want to stop. You can determine the task ID by using the LISTTASK command.
NODETYPE	<p>When CMD=LISTLOCATIONS, specify one of the following node types:</p> <ul style="list-style-type: none"> - N. List locations that are defined in NODE statements in the DBMOVER configuration file. - A. List locations that are defined in NODE or SVCNODE statements in the DBMOVER configuration file. - S. List locations that are defined in SVCNODE statements in the DBMOVER configuration file. <p>Default is N.</p>
LOC	The remote location where the task is running or where the allocated data sets exist. Locations must be specified in NODE statements in the DBMOVER configuration file. If you enter LOCAL, the utility returns an error message.
UID	<p>A user ID that has the authority to access the location, if required by your security settings.</p> <p>For a location on a supported Linux, UNIX, or Windows system, if you have enabled PowerExchange LDAP user authentication, the user name is the enterprise user name. For more information, see the <i>PowerExchange Reference Manual</i>.</p>
PWD	<p>A password or encrypted password for the specified user. If a password contains nonalphanumeric characters, you must enclose it in double quotation marks (""). Do not include double quotation marks within a password string, even if you enclose it in double quotation marks.</p> <p>For access to a remote i5/OS or z/OS location, you can enter a valid PowerExchange passphrase instead of a password. An i5/OS passphrase can be from 9 to 31 characters in length. A z/OS passphrase can be from 9 to 128 characters in length. Passphrases can contain the following characters:</p> <ul style="list-style-type: none"> - Uppercase and lowercase letters - The numbers 0 to 9 - Spaces - The following special characters: ' - ; # \ , . / ! % & * () _ + { } : @ < > ? <p>Note: The first character is an apostrophe.</p> <p>Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.</p> <p>Note: On z/OS, a valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.</p> <p>If a passphrase contains spaces, enclose it in double quotation marks (""). If a passphrase contains special characters, enclose it in triple double-quotation marks ("").</p> <p>To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the <i>PowerExchange Reference Manual</i>.</p>

Example Output:

```
2003-06-27 14:20:25                                TASK LIST

Name          Taskid      Partner      Port Status      Acc Method
=====
                0740                Active      RPX
                1688                Active      TASK_CNTL
```

Displaying DTLUTSK Utility Help

On Linux, UNIX, or Windows, if you enter the DTLUTSK command without parameters or with a question mark (?) only, the utility displays help information on the correct syntax for each CMD operation.

For example:

```
C:\Informatica\PowerExchange10.2.0HF2>dtlutsk ?
DTLUTSK Help: DTLUTSK CMD=LISTTASK/STOPTASK/LISTLOCATIONS/LISTALLOC LOC=location UID=uid
PWD=pwd/EPWD=encryptpwd
DTLUTSK Help: Examples:
DTLUTSK Help: DTLUTSK CMD=LISTTASK LOC=NODE1 UID=uid PWD=pwd
DTLUTSK Help: DTLUTSK CMD=STOPTASK TASKID=taskid LOC=NODE1 UID=uid PWD=pwd
DTLUTSK Help: DTLUTSK CMD=STOPTASK APPL=taskname LOC=NODE1 UID=uid PWD=pwd
DTLUTSK Help: DTLUTSK CMD=LISTLOCATIONS
DTLUTSK Help: DTLUTSK CMD=LISTLOCATIONS LOC=NODE1 NODETYPE=N
DTLUTSK Help: DTLUTSK CMD=LISTALLOC LOC=NODE1 UID=uid PWD=pwd
```

DTLUTSK Job on z/OS

To perform DTLUTSK operations on z/OS, configure JCL statements for the DTLUTSK job and then submit the JCL.

Example JCL for a DTLUTSK Job on z/OS

The following JCL statements are for a DTLUTSK job on z/OS:

```
/*
/* MEMBER DTLUTSK
/*
/*INCS1 INCLUDE MEMBER=GENBULK
/*
/*RUN EXEC PGM=DTLUTSK,
/* PARM=('CMD=LISTTASK LOC=location UID=userid PWD=password')
/*
/* SAMPLE PARMS FOLLOW:
/* REMOVE COMMENT BEFORE CMD TO RUN
/* DTLUTSK Help: Examples:
/* PARM=('CMD=LISTTASK LOC=NODE1 UID=uid PWD=pwd')
/* PARM=('CMD=STOPTASK TASKID=taskid LOC=NODE1 UID=uid PWD=pwd')
/* PARM=('CMD=STOPTASK APPL=taskname LOC=NODE1 UID=uid PWD=pwd')
/* PARM=('CMD=LISTTASK TASKID=taskid LOC=location',
/* 'UID=uid EPWD=encryptpwd')
/*STEPLIB DD DISP=SHR,DSN=&HLQ..LOADLIB
/* DD DISP=SHR,DSN=&SCERUN
/*
/*SYSIN DD DUMMY
/*
/*
/*DTLMSG DD DSN=&HLQ..DTLMSG,DISP=SHR
/* IF USING MESSAGE OVERRIDE THEN CUSTOMIZE BELOW
/**DTLMSG DD DISP=SHR,DSN=&RUNLIB(DTLMSGO)
/*DTLCFG DD DSN=&RUNLIB(DBMOVER),DISP=SHR
```

```
//DTLKEY DD DSN=&RUNLIB(LICENSE),DISP=SHR
//DTLSGN DD DSN=&RUNLIB(SIGNON),DISP=SHR
//DTLLOG DD SYSOUT=*
//DTLLOG01 DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//CEEDUMP DD SYSOUT=*
```

To run this JCL, add a JOB card. You can copy the JOBCARD member to the DTLUTSK member.

In the PARM statements, specify the utility commands and parameters. You can include the following common parameters:

LOC

A node name for the remote location where the task or tasks are running. This node name must be specified in a NODE statement in the DBMOVER configuration file.

UID

A user ID that can be used to access the remote location. You must also specify either the PWD or EPWD parameter.

PWD

A password for the specified user or a valid PowerExchange passphrase.

A passphrase for z/OS access can be from 9 to 128 characters in length and can contain the following characters:

- Uppercase and lowercase letters
- The numbers 0 to 9
- Spaces
- The following special characters:

```
' - ; # \ , . / ! % & * ( ) _ + { } : @ | < > ?
```

Note: The first character is an apostrophe.

Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.

If a passphrase contains spaces, you must enclose it with double-quotation marks ("), for example, "This is a passphrase". If a passphrase contains special characters, you must enclose it with triple double-quotation characters ("""), for example, """This passphrase contains special characters ! % & * . """ . If a passphrase contains only alphanumeric characters without spaces, you can enter it without delimiters.

To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the *PowerExchange Reference Manual*.

Note: On z/OS, a valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.

Do not also specify the EPWD parameter.

EPWD

An encrypted password for the specified user.

For a location on z/OS, you can enter an encrypted PowerExchange passphrase instead of an encrypted password. Do not encrypt a passphrase that contains invalid characters, such as double-quotation marks, single quotation marks, or currency symbols.

Example Output from a DTLUTSK Job on z/OS

```
***** TOP OF DATA *****
2007-10-25 13:28:45
0Name          Taskid      Partner      TASK LIST
Acc_Method      Sessid      Port Status
=====
=====
                x 0001      10.3.4.57      6900 Active      CAPXRT
                0002      127.0.0.1      6900 Active      TASK_CNTL
***** BOTTOM OF DATA *****
```

DTLUTSK Command Line Utility on z/OS

You can issue the following types of commands from the DTLUTSK command line:

- **LISTTASK.** Lists all current tasks.
- **STOPTASK.** Stops the task that is specified in the TASKID parameter.
- **LISTLOCATIONS.** Lists all current locations.
- **LISTALLOC.** Lists all allocated data sets.
- **FREEALLOC.** Frees the allocated data sets that are specified by a DDNAME and data set name.

LISTTASK Command

The LISTTASK command lists all current tasks.

Use the MVS MODIFY command to issue the command.

Syntax:

```
MODIFY listener_name,LISTTASK
```

Example output:

```
PWX-00711 Active tasks:
PWX-00712 Task=task_id, Partner=IP_address, Port=port_number, PwrCntrSess= ,
Application= , Status=task_status, AM=access_method, Mode= , Process= , SessId=
PWX-00713 number active tasks
```

STOPTASK Command

The STOPTASK command stops the task that is specified by the TASKID parameter or by an application name.

Use the MVS MODIFY command to issue the command.

Syntax for stopping by TASKID:

```
MODIFY listener_name,STOPTASK TASKID=taskid
```

Syntax for stopping by application name:

```
MODIFY listener_name,STOPTASK application_name
```

Note: When you stop CDC sessions, STOPTASK waits for a commit boundary before terminating the task. For more information about commit boundaries and processing, see *PowerExchange CDC Guide for z/OS*.

LISTLOCATIONS Command

The LISTLOCATIONS command lists all current locations.

Use the MVS MODIFY command to issue the command.

Syntax:

```
MODIFY listener_name,LISTLOCATIONS [NODETYPE={N|A|S}]
```

LISTALLOC Command

The LISTALLOC command lists all allocated data sets.

Use the MVS MODIFY command to issue the command.

Note: Instead of issuing the LISTALLOC command from the command line, you can specify the LISTALLOC command in the **Pre SQL** or **Post SQL** attribute for a PowerCenter session to report file allocations when the workflow runs. Also specify the PowerCenter **Retrieve PWX Log Entries** connection attribute to enable the informational messages with file-allocation information to be written the session log.

Syntax:

```
MODIFY listener_name,LISTALLOC
```

Example output:

```
Alloc: DDN=<STEPLIB > DSN=<CEE.SCEERUN >
Alloc: DDN=< > DSN=<DTLUSR.DEVB LD.LOADLIB >
Alloc: DDN=< > DSN=<DTLUSR.DEVB LD.LOAD >
Alloc: DDN=< > DSN=<DTLUSR.DEVB LD.NIML.USERLIB >
Alloc: DDN=<DTLAMCPR> DSN=<DTLUSR.DEVB LD.V1.CCT >
Alloc: DDN=<DTLCACDE> DSN=<DTLUSR.DEVB LD.V1.CDEP >
Alloc: DDN=<DTLCACDC> DSN=<DTLUSR.DEVB LD.V1.CDCT >
Alloc: DDN=<DTLCAMAP> DSN=<DTLUSR.DEVB LD.V1.DTLCAMAP >
Alloc: DDN=<DTLMSG > DSN=<DTLUSR.DEVB LD.DTLMSG >
Alloc: DDN=<DTLCFG > DSN=<DTLUSR.V811.RUNLIB >
Alloc: DDN=<DTLKEY > DSN=<DTLUSR.V811.RUNLIB >
Alloc: DDN=<DTLSGN > DSN=<DTLUSR.V811.RUNLIB >
Alloc: DDN=<DTLLOG > DSN=<DTLUSR.DTLLOG.LOG >
Alloc: DDN=<DATAMAP > DSN=<DTLUSR.V811.V1.DATAMAPS >
Alloc: DDN=<SYSUDUMP> DSN=<DTLUSR.DTLUSR2.JOB05761.D0000101.? >
Alloc: DDN=<SYSOUT > DSN=<DTLUSR.DTLUSR2.JOB05761.D0000102.? >
Alloc: DDN=<URLEOUT > DSN=<DTLUSR.DTLUSR2.JOB05761.D0000103.? >
Alloc: DDN=<SYSPRINT> DSN=<DTLUSR.DTLUSR2.JOB05761.D0000104.? >
Alloc: DDN=<CEEDUMP > DSN=<DTLUSR.DTLUSR2.JOB05761.D0000105.? >
Alloc: DDN=<CXX > DSN=<DCOM.V10.CXX >
Alloc: DDN=<DTLOUT > DSN=<DTLUSR.DTLUSR2.JOB05761.D0000106.? >
Alloc: DDN=<DTLERR > DSN=<DTLUSR.DTLUSR2.JOB05761.D0000107.? >
Command < LISTALLOC> succeeded
```

FREEALLOC Command

The FREEALLOC command closes and deallocates a data set that was already dynamically allocated by a PowerExchange Listener.

Use this command in situations such as when a task abends and the resource managers fail to close all the dynamically allocated data sets.

To issue the command, use the MVS MODIFY command.

Syntax:

```
MODIFY listener_name,FREEALLOC DDN=ddname FN=data_set_name
```

The DDN and FN parameters are required.

Note: The data set is not deallocated under any of the following circumstances:

- The request was not from a PowerExchange Listener.
- The request was for a file that has not been dynamically allocated by a Listener and that does not have a DDN name that starts with SYS0.
- The request was issued without a FN file or data set name.

Running the DTLUTSK Utility in the PowerExchange Navigator

The TASK_CNTL data access method is available so that you can perform a database row test to retrieve the results of the LISTTASK, STOPTASK, or LISTLOCATIONS command from the PowerExchange Navigator.

Note: STOPTASK works only with the CAPXRT access method.

To run the DTLUTSK utility in the PowerExchange Navigator:

1. In the **Resource Explorer**, double-click a data map that is defined for the location where the PowerExchange Listener is running to open the data map.
Note: Alternatively, you can open an extraction map or personal metadata profile that is defined for the PowerExchange Listener location.
2. On the **Data Map** tab, select a table view, and then click **File > Database Row Test** on the menu bar. A message might prompt you to send the data map to a remote location.
3. In the **Data Map Remote Node** dialog box, enter connection information for the location where the PowerExchange Listener is running, and click **OK**.
The **Database Row Test** dialog box appears.
4. In the **DB Type** list, select **TASK_CNTL**.
5. In the **Fetch** list, select one of the following commands:
 - **List Locations.** Displays information about the locations that are defined in NODE or SVCNODE statements in the DBMOVER configuration file on the system where the PowerExchange Listener is running. The output includes the node name, IP address, port number, send and receive buffer sizes and lengths, receive timeout, and SSL use.
 - **List Task.** Displays information about each active task that is running under the PowerExchange Listener. The output includes the task ID, TCP/IP address, port number, application name, access type, and status.
 - **Stop Task.** Stops a specific PowerExchange Listener task. To identify the task, you must enter a task ID or application name.
6. If you are issuing a STOPTASK command, enter a task ID or application name in the **SQL Statement** box. Use the following syntax:

```
stoptask {taskid=task_id|appname=application_name}
```

Do not include the curly brackets. These brackets indicate a choice of either taskid or appname is required.

Note: Do not enter the application name in the **Application** field. The **Application** field is ignored for TASK_CNTL commands.

7. If you are issuing a LISTLOCATIONS command, optionally enter the node type in the **SQL Statement** box. Use the following syntax:

```
listlocations nodetype={N|A|S}
```

Specify one of the following values for nodetype:

- N. Default. List locations that are defined in NODE statements in the DBMOVER file.
- A. List locations that are defined in NODE or SVCNODE statements in the DBMOVER file.
- S. List locations that are defined in SVCNODE statements in the DBMOVER file.

8. Click **Go**.

The **Database Row Test Output** window displays the output for the command.

For more information, see the "Database Row Test" chapter in the *PowerExchange Navigator User Guide*.

DTLUTSK Utility Security

You can configure security for the DTLUTSK utility. The strategy that you use depends on the utility platform.

DTLUTSK Utility Security Requirements on z/OS

In the DBMOVER configuration file, set the SECURITY statement is set to (2,x). Also define the RACF_CLASS statement to control access to the utility by using a class name that is defined in RACF or similar z/OS security package.

Also grant the appropriate access to the required users. For example:

```
DTL.TASKCTRL.DISPLAY
DTL.TASKCTRL.STOPTASK
```

These statements enable users to display active tasks and to stop tasks.

DTLUTSK Utility Security Requirements on i5/OS

On i5/OS, if the SECURITY parameter is set to (2,x), where x is N or Y, you must define security statements as follows, replacing DATALIB with the required data library:

```
GRTOBJAUT OBJ(DATALIB/AUTHSKLST) OBJTYPE(*FILE) USER(USERID) AUT(*USE)
GRTOBJAUT OBJ(DATALIB/AUTHSKSTP) OBJTYPE(*FILE) USER(USERID) AUT(*USE)
```

Using Signon.txt to Authorize Users to Display or Stop Tasks

If running with a configuration setting of SECURITY=(n,Y) where n is 0 to 2, an additional parameter is available for allowing the use of list and stop tasks:

```
/* 4. TASKCNTRL= is an optional function allowed
/* Format is D or S
/* If it is supplied, then the user can use Task Control to
/* Display or Stop tasks.
/* This signon list will only be used if Security=(n,Y) is used
/* in the config.
```

CHAPTER 15

EDMLUCTR - Log Scan and Print Utility

This chapter includes the following topics:

- [EDMLUCTR Utility Overview, 218](#)
- [Supported Operating Systems for the EDMLUCTR Utility, 218](#)
- [Control Statement Syntax for the EDMLUCTR Utility, 219](#)
- [Control Statement Parameters for the EDMLUCTR Utility, 219](#)
- [Running the EDMLUCTR Utility, 221](#)
- [EDMLUCTR Utility Usage Notes, 221](#)
- [EDMLUCTR Utility Examples, 221](#)

EDMLUCTR Utility Overview

Use the EDMLUCTR utility to perform the following tasks:

- Produce summary information about each log record.
- Produce detailed information about change records and units of work (UOWs) records.
- Produce summary information, by registration tag name, about all sources for which changes are captured.
- List UOWs that have not yet ended.

For more information about the PowerExchange Logger and Post Log Merge, see *PowerExchange CDC Guide for z/OS*.

Supported Operating Systems for the EDMLUCTR Utility

The EDMLUCTR utility can run on z/OS only.

Control Statement Syntax for the EDMLUCTR Utility

Use the following syntax for the EDMLUCTR utility control statements:

```
[ -SEL  
    [CHANGE-DETAIL]  
    [LOGRBA=logrba]  
    [ENDRBA=endrba]  
    [PACKET-DETAIL]  
    [RECORDS={ nnnnnnnn | EOF } ]  
    [SUMM] ]  
  
[ -MASK mask ]
```

The following rules and guidelines apply:

- Use the SYSIN DD JCL statement to enter the utility control statements.
- All of the control statements are optional and begin in column 1.
- Control statements must end with a blank and must not exceed 80 characters in length.
- Use one or more spaces as a delimiter between parameters for a control statement.
- No continuation syntax exists.
- If more than a single line is required for a -SEL control statement, code -SEL at the beginning of each subsequent line that includes additional parameters.
- The value for a parameter cannot continue from line to line.
- If you code multiple -MASK statements, only the last one is used.

Control Statement Parameters for the EDMLUCTR Utility

Review the parameter descriptions to determine which parameters to use in the EDMLUCTR control statements.

-SEL Statement

-SEL has the following parameters:

CHANGE-DETAIL

Optional. Prints summary and detailed information, in hexadecimal format, about change records. If not specified, only summary information for change records prints.

LOGRBA

Optional. Specifies an RBA in the log data sets to use as the starting point for the EDMLUCTR utility. In a Post-Log Merge environment, LOGRBA specifies a timestamp value in the log data sets as an unstructured TOD-clock value.

As the starting point, EDMLUCTR uses the first log record that has an RBA or a timestamp that is equal to or higher than the specified value.

Specify up to 12 hexadecimal digits for the LOGRBA value. You can omit leading zeroes.

Note: With Post-Log Merge configurations, LOGRBA must be specified and the LOGRBA value must be 16 hexadecimal digits. LOGRBA values represent the timestamp of the requested data when using Post-Log Merge.

If no parameter is specified, LOGRBA is the default. Its default value is the RBA that is recorded in the emergency restart data set (ERDS) from the latest checkpoint.

ENDRBA

Optional. Specifies an RBA in the log data sets to use as the ending point for the EDMLUCTR utility. In a Post-Log Merge environment, ENDRBA specifies a timestamp value in the log data sets as an unstructured TOD-clock value.

EDMLUCTR prints or scans log records until it finds a log record that has an RBA or a timestamp that is equal to or greater than the specified ENDRBA value. When EDMLUCTR reaches that point, it ends.

Specify up to 16 hexadecimal digits for the ENDRBA value. You can omit leading zeroes.

PACKET-DETAIL

Optional. Prints summary and detailed information, in hexadecimal format, about UOW records. If not specified, only summary information for UOW records prints.

RECORDS

Optional. Prints or scans the specified number of log records.

When you specify RECORDS=EOF, EDMLUCTR prints all records from the specified or default start location to the current end of the log data.

If you specify -SEL RECORDS and the -MASK statement, EDMLUCTR uses the RECORDS value as the number of records to scan for the mask value rather than as the number of records to print.

Minimum is 1. Maximum is 99999999. Default is 5,000.

SUMM

Optional. Prints only change summary information.

Change summary information includes the total number of inserts, updates, and deletes found in the log data scanned, ordered by source registration tag name.

-MASK Statement

-MASK has the following parameter:

mask

Required. Specify a filter in one of the following formats:

- A character value, such as a DB2 table name, without embedded blanks. Use the hexadecimal format for character strings with embedded blanks.
- A hexadecimal value, such as a UOW number. Enclose hexadecimal character strings in single quotes and precede the string with the letter X.

If you specify both -SEL RECORDS and -MASK, EDMLUCTR uses the RECORDS value as the number of records to scan for the mask value rather than as the number of records to print.

Maximum length is 70 characters.

Running the EDMLUCTR Utility

PowerExchange provides sample JCL for the EDMLUCTR utility in the LOGPRINT member of the SAMPLIB library.

The following JCL statements are required to run the utility:

```
//          JOB
//READER    EXEC  PGM=EDMLUCTR
//STEPLIB   DD    DISP=SHR,DSN=hlq.LOAD
//ERDS01    DD    DISP=SHR,DSN=your.ERDS01
//EDMPARMS  DD    DISP=SHR,DSN=your.USERLIB
//SYSIN     DD    *
```

JOB

Initiates the job.

EXEC PGM=EDMLUCTR

Invokes the utility.

STEPLIB DD

Defines the PowerExchange LOAD library that contains the utility.

ERDS01 DD

Defines the PowerExchange Logger emergency restart data set (ERDS) that contains the inventory of log data sets containing the log records to be displayed. Specify only one ERDS data set.

EDMPARMS DD

Defines the data set that contains the EDMSDIR options module.

SYSIN DD

Defines the utility control statements.

EDMLUCTR Utility Usage Notes

Consider the following points before using the EDMLUCTR utility:

- If you specify old LOGRBA values, the utility might read archive log data sets that have been migrated by the storage management system. Verify that you have sufficient DASD to recall any migrated archive log data sets.
- You can use the EDMLUCTR utility in either a single PowerExchange Logger environment or Post-Log Merge environment.
- You can run the EDMLUCTR utility whether or not the PowerExchange Logger is running.

EDMLUCTR Utility Examples

The following are examples of the EDMLUCTR utility.

EDMLUCTR Utility - Example 1

The following JCL statements print summary data for all log records, starting with the RBA recorded in the ERDS at the time the latest PowerExchange Logger checkpoint was taken:

```
// JOB
//READER EXEC PGM=EDMLUCTR
//STEPLIB DD DISP=SHR,DSN=hlq.LOAD
//ERDS01 DD DISP=SHR,DSN=your.ERDS01
//EDMPARMS DD DISP=SHR,DSN=your.USERLIB
//SYSIN DD *
//
```

The resulting output is:

```
18:53:31.86 LOG START
PXWEDM175000I Log Scan/Print Utility initializing
Echo of input from SYSIN.....
End of input from SYSIN.....
PXWEDM175005I Begin data transfer at X'0000000050000000'
PXWEDM172146I EDMLRDP: LMF now processing EDMTEST.DEV.V1.PRLOG.DS01 for Log Scan/Print Utility
Log-rec EDP-UOW=LOGGER00000000500000000000 LogRBA=0000000050000000
Log-rec EDP-UOW=LOGGER0000000050B400000001 LogRBA=0000000050B40000
Log-rec EDP-UOW=LOGGER00000000523400000003 LogRBA=0000000052340000
Log-rec EDP-UOW=AUSL 0000000052E800000001 LogRBA=0000000052E80000
Beg-pkt EDP-UOW=AUSL 00000000546800000000 LogRBA=0000000054680000
ECCR-UOW=AUSPRT01 AUSPRT01 C1E4E2D7D9E3F0F1 008F803000000001
Timestamp-18:35:40:11 Date-04/29/2014
Chg-rec EDP-UOW=AUSL 00000000546800000000 LogRBA=0000000054F00000
ECCR-UOW=AUSPRT01 C1E4E2D7D9E3F0F1 008F803000000001
Source=VSM Func=ISRT Srcname=VSAMEDMTEST.VSAM.KSDS01
Timestamp-18:35:40:18 Date-04/29/2014
Chg-rec EDP-UOW=AUSL 00000000546800000000 LogRBA=0000000056550000
ECCR-UOW=AUSPRT01 C1E4E2D7D9E3F0F1 008F803000000001
Source=VSM Func=ISRT Srcname=VSAMEDMTEST.VSAM.KSDS01
Timestamp-18:35:40:36 Date-04/29/2014
. . . . .
. . . . .
. . . . .
Srv-rec EDP-UOW=ECCRCTF5F2404040404040404 LogRBA=0000001902F90000
Srv-rec EDP-UOW=ECCRCTF5F2404040404040404 LogRBA=0000001906090000
Log-rec EDP-UOW=AUSL 00000019093500000001 LogRBA=0000001909350000
Srv-rec EDP-UOW=ECCRCTF5F2404040404040404 LogRBA=000000190AB50000
Srv-rec EDP-UOW=ECCRCTF5F2404040404040404 LogRBA=000000190DC50000
Log-rec EDP-UOW=AUSL 00000019110500000001 LogRBA=0000001911050000
Srv-rec EDP-UOW=AUSDB2F0F10000000000004040 LogRBA=0000001912850000
Srv-rec EDP-UOW=AUSDB2F0F10000000000004040 LogRBA=0000001916890000
Beg-pkt EDP-UOW=AUSL 000000191A8D00000000 LogRBA=000000191A8D0000
ECCR-UOW=AUSDB201 01 F0F1000000000001 605FE82B00000000
Timestamp-18:53:19:68 Date-05/07/2014
Chg-rec EDP-UOW=AUSL 000000191A8D00000000 LogRBA=000000191D410000
ECCR-UOW=01 F0F1000000000001 605FE82B00000000
Source=DB2 Func=DLET Srcname=DB2DSNBtenchar1
Timestamp-18:53:19:68 Date-05/07/2014
Chg-rec EDP-UOW=AUSL 000000191A8D00000000 LogRBA=00000019219F0000
ECCR-UOW=01 F0F1000000000001 605FE82B00000000
Source=DB2 Func=DLET Srcname=DB2DSNBtenchar1
Timestamp-18:53:19:68 Date-05/07/2014
Com-pkt EDP-UOW=AUSL 000000191A8D00000000 LogRBA=0000001925E90000
ECCR-UOW=AUSDB201 01 F0F1000000000001 605FE82B00000000
Timestamp-18:53:19:68 Date-05/07/2014
PXWEDM172198I EDMLPOPU: LMF table populate tasks are terminating due to an operator stop or PAC termination
PXWEDM175039E Reader for log_dataset_name ended due to operator STOP or PAC termination
Totals by Source.....
VSAMEDMTEST.VSAM.KSDS01 Isrt= 5 Repl= 20 Dlet= 0 Segm= 0 Evnt= 0 Unk= 0
VSAMEDM.DEV.EDMAB123 Isrt= 0 Repl= 0 Dlet= 0 Segm= 0 Evnt= 0 Unk= 0
VSAMEDM.DEV.EDMAB123 Isrt= 1000 Repl= 0 Dlet= 1000 Segm= 0 Evnt= 0 Unk= 0
DB2DSNBtenchar1 Isrt= 2 Repl= 2 Dlet= 2 Segm= 0 Evnt= 0 Unk= 0
Open Uows.....
LOG END
```

If the change records are segmented, the EDMLUCTR utility shows chaining information in the Chg-rec and Seg-rec fields and counts only the first record in the totals for Isrt, Repl, and Dlet fields. For example:

```
PXWEDM172146I EDMLRDP: LMF now processing EDMTEST.DEV.V1.PRLOG.DS01 for Log Scan/Print Utility
...
Chg-rec EDP-UOW=QAJL 0000DD53D41400000000 LogRBA=0000DD53D5B20000
ECCR-UOW=QAA826ME D8C1C1F8F2F6D4C5 00000000000010CA
Chaining FLG=80 SEQ=01 LEN=0000F63E
Source=VSM Func=UPDT Srcname=ABCABC8261DBID08261FILEID00352
Timestamp-11:34:23:52 Date-08/10/2017
Seg-rec EDP-UOW=QAJL 0000DD53D41400000000 LogRBA=0000DD5456360000
ECCR-UOW=QAA826ME D8C1C1F8F2F6D4C5 00000000000010CA
Chaining FLG=20 SEQ=02 LEN=0000F63E
```

```

Source=VSM Func=ISRT Srcname=ABCABC8261DBID08261FILEID00352
Timestamp=11:34:23:52 Date=08/10/2017
...
Totals by Source.....
IDLQADLGSIDstunflat1      Isrt= 4285      Repl= 20      Dlet= 0      Segm= 0      Evnt= 0      Unk= 0
EDM_BASEEDM_EVENT_MARKER  Isrt= 0        Repl= 0      Dlet= 0      Segm= 0      Evnt= 28     Unk= 0
ABCABC8261DBID08261FILEID00312  Isrt= 2        Repl= 16     Dlet= 10     Segm= 0      Evnt= 0      Unk= 0
ABCABC8261DBID08261FILEID00352  Isrt= 16       Repl= 29     Dlet= 5      Segm= 29     Evnt= 0      Unk= 0
ABCABC8261DBID08261FILEID00313  Isrt= 2        Repl= 2      Dlet= 2      Segm= 0      Evnt= 0      Unk= 0
ABCABC8261DBID08261FILEID00355  Isrt= 29       Repl= 7      Dlet= 22     Segm= 260    Evnt= 0      Unk= 0
ABCABC8261DBID08261FILEID00351  Isrt= 56       Repl= 98     Dlet= 56     Segm= 1232   Evnt= 0      Unk= 0
IDLQADLGSIDstundent1      Isrt= 2861     Repl= 11109  Dlet= 0      Segm= 0      Evnt= 0      Unk= 0
IDLQADLGSIDcoursen1      Isrt= 1061     Repl= 18101  Dlet= 0      Segm= 0      Evnt= 0      Unk= 0
IDLQADLGSIDcrsenmem1      Isrt= 18100    Repl= 39272  Dlet= 0      Segm= 0      Evnt= 0      Unk= 0

```

EDMLUCTR Utility - Example 2

The following JCL statements print summary data for all log records starting from the specified RBA and detailed information in hexadecimal format about the change records:

```

//          JOB
//READER    EXEC PGM=EDMLUCTR
//STEPLIB   DD   DISP=SHR,DSN=hlq.LOAD
//ERDS01    DD   DISP=SHR,DSN=your.ERDS01
//EDMPARMS  DD   DISP=SHR,DSN=your.USERLIB
//SYSIN     DD   *
-SEL LOGRBA=000000191A8D0000 CHANGE-DETAIL
-SEL RECORDS=10
//

```

Note: The job prints the detailed information about the change records because the optional CHANGE-DETAIL parameter is included.

The resulting output is:

```

19:08:00.74 LOG START
PXWEDM175000I Log Scan/Print Utility initializing
Echo of input from SYSIN.....
-SEL LOGRBA=000000191A8D0000 CHANGE-DETAIL
-SEL RECORDS=10
End of input from SYSIN.....
PXWEDM175005I Begin data transfer at X'000000191A8D0000'
PXWEDM172146I EDMLRDP: LMF now processing WBRUMB1.DEV.V1.PRILOG.DS01 for Log Scan/Print Utility
  Beg-pkt  EDP-UOW=AUSL 000000191A8D00000000 LogRBA=000000191A8D0000
            ECCR-UOW=AUSDB201 01 F0F1000000000001 605FE82B00000000
            Timestamp=18:53:19:68 Date=05/07/2014
  Chg-rec  EDP-UOW=AUSL 000000191A8D00000000 LogRBA=000000191D410000
            ECCR-UOW=01 F0F1000000000001 605FE82B00000000
            Source=DB2 Func=DLET Srcname=DB2DSNBtenchar1
            Timestamp=18:53:19:68 Date=05/07/2014
            0000 00000003 00000014 000002B4 000002B4 © ©
            0010 0000044A 02A00101 C3C46008 00000002 æ µ CD--
            0020 E2C40000 00000000 00000000 00040000 SD æ
            0030 00000000 00000000 0000CD1E 42982685 ò â q e
            0040 8081C4C2 F2C4E2D5 C2A38595 83888199 @aDB2DSNBtenchar
            0050 F1404040 40404040 40404040 40404040 l
            0060 40404040 4040C1E4 E2D34040 00000019 AUSL
            0070 1A8D0000 00000000 00191D41 00000000 'y
. . . . .
Default 5000 or RECORDS= threshold reached
PXWEDM172198I EDMLPOPU: LMF table populate tasks are terminating due to an operator stop or PAC termination
PXWEDM175039E Reader for log_dataset_name ended due to operator STOP or PAC termination
Totals by Source.....
          DB2DSNBtenchar1      Isrt= 2      Repl= 1      Dlet= 2      Segm= 0      Evnt= 0      Unk= 0
Open Uows.....
      Edp-UOW=AUSL 0000001936AD00000000 LogRBA=0000001936AD0000 ECCR-UOW=01 F0F1000000000001 605FF6E000000000
LOG END

```

EDMLUCTR Utility - Example 3

The following JCL statements filter records by using the -MASK value of DB2DSNB and then print the records starting from a specific RBA.

```

//          JOB
//READER    EXEC PGM=EDMLUCTR
//STEPLIB   DD   DISP=SHR,DSN=hlq.LOAD

```

```
//ERDS01 DD DISP=SHR,DSN=your.ERDS01
//EDMPARMS DD DISP=SHR,DSN=your.USERLIB
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
-SEL LOGRBA=000000191A8D0000 RECORDS=10
-MASK DB2DSNB
//
```

The inclusion of the optional RECORDS parameter limits the number of record scans for the character string DB2DSNB.

The resulting output is:

```
19:12:40.93 LOG START
PWXEDM175000I Log Scan/Print Utility initializing
Echo of input from SYSIN.....
-SEL LOGRBA=000000191A8D0000 RECORDS=10
-MASK DB2DSNB
End of input from SYSIN.....
PWXEDM175005I Begin data transfer at X'000000191A8D0000'
PWXEDM172146I EDMLRDP: LMF now processing WBRUMB1.DEV.V1.PRILOG.DS01 for Log Scan/Print Utility
Chg-rec EDP-UOW=AUSL 000000191A8D00000000 LogRBA=000000191D410000
        ECCR-UOW=01 F0F1000000000001 605FE82B00000000
        Source=DB2 Func=DLET Srcname=DB2DSNBtenchar1
        Timestamp=18:53:19:68 Date=05/07/2014
Chg-rec EDP-UOW=AUSL 000000191A8D00000000 LogRBA=00000019219F0000
        ECCR-UOW=01 F0F1000000000001 605FE82B00000000
        Source=DB2 Func=DLET Srcname=DB2DSNBtenchar1
        Timestamp=18:53:19:68 Date=05/07/2014
Chg-rec EDP-UOW=AUSL 00000019289D00000000 LogRBA=000000192B510000
        ECCR-UOW=01 F0F1000000000001 605FF25800000000
        Source=DB2 Func=ISRT Srcname=DB2DSNBtenchar1
        Timestamp=18:53:19:68 Date=05/07/2014
Chg-rec EDP-UOW=AUSL 00000019289D00000000 LogRBA=000000192F9B0000
        ECCR-UOW=01 F0F1000000000001 605FF25800000000
        Source=DB2 Func=ISRT Srcname=DB2DSNBtenchar1
        Timestamp=18:53:19:68 Date=05/07/2014
Chg-rec EDP-UOW=AUSL 0000001936AD00000000 LogRBA=0000001939610000
        ECCR-UOW=01 F0F1000000000001 605FF6E000000000
        Source=DB2 Func=UPDT Srcname=DB2DSNBtenchar1
        Timestamp=18:53:19:68 Date=05/07/2014
Default 5000 or RECORDS= threshold reached
PWXEDM172198I EDMLPOPU: LMF table populate tasks are terminating due to an operator stop or PAC termination
PWXEDM175039E Reader for log_dataset_name ended due to operator STOP or PAC termination
Totals by Source.....
        DB2DSNBtenchar1 Isrt= 2 Repl= 1 Dlet= 0 Segm= 0 Evnt= 0 Unk= 0
Open Uows.....
LOG END
```

EDMLUCTR Utility - Example 4

If you run the utility in a Post-Log Merge environment, the following JCL statements print summary data for all log records starting from a specific timestamp:

```
// JOB
//READER EXEC PGM=EDMLUCTR
//STEPLIB DD DISP=SHR,DSN=hlq.LOAD
//ERDS01 DD DISP=SHR,DSN=your.ERDS01
//EDMPARMS DD DISP=SHR,DSN=your.USERLIB
//SYSIN DD *
-SEL LOGRBA=CD1FCF19F4713301 RECORDS=EOF
//
```

The resulting output is:

```
19:27:31.86 LOG START
PWXEDM175000I Log Scan/Print Utility initializing4
Echo of input from SYSIN.....
-SEL LOGRBA=CD1FCF19F4713301 RECORDS=EOF
End of input from SYSIN.....
PWXEDM175005I Begin data transfer at X'0000000050000000'
PWXEDM172146I EDMLRDP: LMF now processing AUSQA.DEV.V1.PRILOG.DS01 for Log Scan/Print Utility
Log-rec EDP-UOW=LOGGER00000000500000000000 LogRBA=CD1FCF19F4713301
Log-rec EDP-UOW=LOGGER0000000050B400000001 LogRBA=CD1FCF19F472B075
Log-rec EDP-UOW=LOGGER00000000523400000003 LogRBA=CD1FCF19F473A691
Log-rec EDP-UOW=AUSL 0000000052E800000001 LogRBA=CD1FCF19F4751B0A
Beg-pkt EDP-UOW=AUSL 00000000546800000000 LogRBA=CD1FCF19F47644B1
        ECCR-UOW=AUSPRT01 AUSPRT01 E6D9C2D7D9E3F0F1 008F8030000000001
```



```

Timestamp-19:25:40:11 Date-04/29/2014
Chg-rec EDP-UOW=AUSL 00000000546800000000 LogRBA=CD1FCF19F47815C3
        ECCR-UOW=AUSPRT01 E6D9C2D7D9E3F0F1 008F803000000001
        Source=VSM Func=ISRT Srcname=VSAMEDM.VSAM.KSDS01
Timestamp-19:25:40:18 Date-04/29/2014
Chg-rec EDP-UOW=AUSL 00000000546800000000 LogRBA=CD1FCF19F47AC50D
        ECCR-UOW=AUSPRT01 E6D9C2D7D9E3F0F1 008F803000000001
        Source=VSM Func=ISRT Srcname=VSAMEDM.VSAM.KSDS01
Timestamp-19:25:40:36 Date-04/29/2014
Chg-rec EDP-UOW=AUSL 00000000546800000000 LogRBA=CD1FCF19F47B2F04
        ECCR-UOW=AUSPRT01 E6D9C2D7D9E3F0F1 008F803000000001
        Source=VSM Func=ISRT Srcname=VSAMEDM.VSAM.KSDS01
Timestamp-19:25:40:55 Date-04/29/2014
Chg-rec EDP-UOW=AUSL 00000000546800000000 LogRBA=CD1FCF19F47B6D63
        ECCR-UOW=AUSPRT01 E6D9C2D7D9E3F0F1 008F803000000001
        Source=VSM Func=ISRT Srcname=VSAMEDM.VSAM.KSDS01
Timestamp-19:25:40:81 Date-04/29/2014
Chg-rec EDP-UOW=AUSL 00000000546800000000 LogRBA=CD1FCF19F47D3D94
        ECCR-UOW=AUSPRT01 E6D9C2D7D9E3F0F1 008F803000000001
        Source=VSM Func=ISRT Srcname=VSAMEDM.VSAM.KSDS01
Timestamp-19:25:41:03 Date-04/29/2014
PH1-pkt EDP-UOW=AUSL 00000000546800000000 LogRBA=CD1FCF19F47D4581
        ECCR-UOW=AUSPRT01 AUSPRT01 E6D9C2D7D9E3F0F1 008F803000000001
Timestamp-19:25:41:39 Date-04/29/2014
Com-pkt EDP-UOW=AUSL 00000000546800000000 LogRBA=CD1FCF19F47F0362
        ECCR-UOW=AUSPRT01 AUSPRT01 E6D9C2D7D9E3F0F1 008F803000000001
Timestamp-19:25:41:39 Date-04/29/2014
Srv-rec EDP-UOW=AUSPRTF0F14040404040404040 LogRBA=CD1FCF19F47E2049
Srv-rec EDP-UOW=AUSPRTF0F14040404040404040 LogRBA=CD1FCF19F48B60C0
Log-rec EDP-UOW=AUSL 00000000634900000001 LogRBA=CD1FCF19F48F240A

. . . . .
. . . . .
. . . . .

Beg-pkt EDP-UOW=AUSL 0000001936AD00000000 LogRBA=CD1FCF19F48F8030
        ECCR-UOW=AUSDB201 01 F0F1000000000001 605FF6E000000000
Timestamp-19:27:19:68 Date-05/07/2014
Chg-rec EDP-UOW=AUSL 0000001936AD00000000 LogRBA=CD1FCF19F493A0D2
        ECCR-UOW=01 F0F1000000000001 605FF6E000000000
        Source=DB2 Func=UPDT Srcname=DB2DSNBtenchar1
Timestamp-19:27:19:68 Date-05/07/2014
Chg-rec EDP-UOW=AUSL 0000001936AD00000000 LogRBA=CD1FCF19F4950D34
        ECCR-UOW=01 F0F1000000000001 605FF6E000000000
        Source=DB2 Func=UPDT Srcname=DB2DSNBtenchar1
Timestamp-19:27:19:68 Date-05/07/2014
Com-pkt EDP-UOW=AUSL 0000001936AD00000000 LogRBA=CD1FCF19F497F385
        ECCR-UOW=AUSDB201 01 F0F1000000000001 605FF6E000000000
Timestamp-19:27:40:69 Date-05/07/2014
PXWEDM172198I EDMLPOPU: LMF table populate tasks are terminating due to an operator stop or PAC termination
PXWEDM175039E Reader for log_dataset_name ended due to operator STOP or PAC termination
Totals by Source.....
        VSAMEDM.VSAM.KSDS01 Isrt= 5 Repl= 0 Dlet= 0 Segm= 0 Evnt= 0 Unk= 0
        VSAMEDM.QA.EDMABC04 Isrt= 0 Repl= 1000 Dlet= 0 Segm= 0 Evnt= 0 Unk= 0
        VSAMEDM.QA.EDMABC07 Isrt= 1000 Repl= 0 Dlet= 1000 Segm= 0 Evnt= 0 Unk= 0
        DB2DSNBtenchar1 Isrt= 2 Repl= 2 Dlet= 2 Segm= 0 Evnt= 0 Unk= 0
Open Uows.....
L O G E N D

```

CHAPTER 16

EDMXLUTL - Event Marker Utility

This chapter includes the following topics:

- [EDMXLUTL Utility Overview, 226](#)
- [Creating an Event Marker in Batch Mode, 226](#)
- [EDMXLUTL Utility JCL Statements, 227](#)
- [EDMXLUTL Utility Control Statements, 227](#)
- [EDMXLUTL Utility EVENT Command, 227](#)
- [Keyword Sets for the BASEEDM Category, 228](#)
- [EDMXLUTL Utility Example, 231](#)

EDMXLUTL Utility Overview

Use the EDMXLUTL utility to create an event marker in your PowerExchange Logger for z/OS.

Creating an Event Marker in Batch Mode

Use the following procedure to create an event marker in batch mode.

To create an event marker in batch mode:

1. Make a working copy of the #EDMLUTB sample JCL from the HLQ.SAMPLIB sample library, where HLQ is the high-level qualifier specified at installation, and edit the copy as required.
2. Run the job to create the event marker.

EDMXLUTL Utility JCL Statements

The following table describes the JCL statements for the EDMXLUTL utility:

Statement	Description
EXEC	Specify the EDMXLUTL program.
STEPLIB DD	Include the PowerExchange Change Capture load library. If you added the load library to your system's LNKLIST concatenation, you do not need to add it to the STEPLIB.
EDMPARMS DD	Specify the name of the user library (YOUR.USERLIB) that contains the default options module (EDMSDIR) associated with the PowerExchange Logger you are using. If you do not include an EDMPARMS DD statement, or if you specify a library that does not contain the options modules, PowerExchange Change Capture uses the STEPLIB concatenation to obtain the configuration options.
EDMSG DD	Specify the data set name to which you want to issue errors and warnings.
EDMSYSIN DD	Specify the appropriate EVENT command for the marker that you want to create.

EDMXLUTL Utility Control Statements

The following table lists the control statements for the event-marker utility:

Command
<pre>EVENT TYPE=BASEEDM NOTIFY={EDITION ENDCOPY COPY} OBJECT={IMS VSAM DB2} ACCESS=STRUCTURE {DBD={dbd_name DSN=data_set_name SYSID=ssid}</pre>
<pre>EVENT TYPE=BASEEDM NOTIFY={EDITION ENDCOPY COPY} OBJECT={IMS VSAM DB2} ACCESS=OBJECT {EDMNAME=edmname DBD=dbd_name} DSN=data_set_name SEGMENT=segment_name [SEGMENT=segment_name ...] DBD=dbd_name DSN=data_set_name SYSID=ssid CREATOR=table_creator TABNAME=table_name [TABNAME=table_name ...] }</pre>

RELATED TOPICS:

- [“EVENT Command Syntax” on page 228](#)
- [“Keyword Sets for the BASEEDM Category” on page 228](#)

EDMXLUTL Utility EVENT Command

Use the EVENT command to create event markers in batch mode.

EVENT Command Syntax

Use the following syntax for the EVENT command:

```
EVENT TYPE=category keyword1=value1 keyword2=value2  
keyword3=value3 ...
```

Subsequent sections discuss the parameters for this command, by category. Each category has one or more sets of keywords associated with it.

EVENT Command Usage

To use this command, include it as a control statement in a batch job. Then, run the job to create the event marker. The following rules apply to specifying this control statement:

- Your statement should be contained within columns 1 through 71.
- If your statement will not fit in this range, you must have a character in column 72 to indicate that your statement continues on more than one line.
- A statement that continues on more than one line must contain only a single command.
- Continued statements must begin in column 1, if column 71 on the previous line is blank.
- A statement can use up to a 38 lines.
- You can use a maximum of 255 blanks to separate commands and keywords.

The following additional information is listed for this command:

- Before you run a job to create an event marker, be sure that the PowerExchange Logger is active.
- A PowerExchange Logger failure could cause the logger to stop while running an event marker job. In this case, the control statements processed prior to the failure are still accepted. Conversely, the control statement that is in progress when the PowerExchange Logger fails, and the subsequent control statements, cause the event marker utility to abend.
- Take care if running this command while the PowerExchange active log is receiving other log records for the source object that the marker affects. This can mix the event marker in with the other records, producing unexpected results.
- When the utility successfully records the event marker record in the PowerExchange log, the utility displays message DTLED175016I. This message provides the RBA of the event marker record within the log. You may need the RBA to reference that record.
- This utility obtains the name of the PowerExchange Logger that it accesses from the default options module, `EDMSDIR`.

Keyword Sets for the BASEEDM Category

Use the BASEEDM category to create a special event record in the PowerExchange active log. This section describes the two keyword sets that you can use with the BASEEDM category:

- MARK
- NOTIFY

MARK Keyword Set

The MARK keyword set tells the event-marker utility to insert a special marker into the PowerExchange Logger for z/OS active logs. The marker returns a log address and passes a signal to a component that uses PowerExchange Logger data.

Note: Use the MARK keyword set only at the direction of Informatica Global Customer Support.

Syntax:

```
EVENT TYPE=BASEEDM MARK=type DATA=text
```

Example:

```
EVENT TYPE=BASEEDM MARK=EOL DATA='my text'
```

The following table describes the keywords that you can use in place of the variable for the MARK statement:

Variable	Keyword Description
<i>type</i>	<p>Tells the utility the type of event marker to add to the log.</p> <p>The following keywords are valid:</p> <ul style="list-style-type: none">- EOD. Creates an event marker that indicates that the end of day has been reached.- SIGNAL. Creates an event marker that indicates a starting point within the log or that passes a signal to a component that uses PowerExchange logger data.- EOL. Creates an event marker that indicates the end of the log. The utility places the marker at the current end of the PowerExchange active log. For the utility to identify the precise end of the log, the PowerExchange Logger should not receive any other records.
<i>text</i>	<p>Enter up to 30 characters of text that you want the utility to add to the event marker record. If you include embedded blanks, you must enclose the text in single quotation marks (').</p>

NOTIFY Keyword Set

This set of keywords tells the utility to insert a special marker into the PowerExchange active log. The special marker notifies the component using the data of an event change, such as a change in the edition value.

This is used to generate a restart point in the PowerExchange Change Capture log.

Syntax:

```
For ACCESS=STRUCTURE:
EVENT TYPE=BASEEDM NOTIFY=type OBJECT=database_type
      ACCESS=STRUCTURE {DBD=database_name DSN=data_set_name |
      SYSID=ssid}
For ACCESS=OBJECT:
EVENT TYPE=BASEEDM NOTIFY=type OBJECT=db_type
      ACCESS=level_of_data_objects
      {EDMNAME=edmtime |
      DBD=database_name DSN=data_set_name SEGMENT=segment_name
      [SEGMENT=segment_name ...] | DBD=database_name DSN=data_set_name |
      SYSID=ssid CREATOR=tbcreator TABNAME=table_name
      [TABNAME=table_name ...]}
```

The following table lists and describes the variables that you can use with the BASEEDM category:

Variables	Description
<i>type</i>	Tells the utility what type of notification the event marker signals. The following value is valid: - EDITION provides notification that a resource registration is changing.
<i>db_type</i>	Indicates the database type of the associated resource. The following values are valid: - IMS - VSAM - DB2
<i>level_of_data_objects</i>	Indicates the level of data objects to be associated with the notification. The following values are valid: - STRUCTURE indicates that all data objects within the database, data set, or subsystem are to be associated with the notification. When you specify ACCESS=STRUCTURE, you must specify either the DBD and data set name or the subsystem ID. For example, for OBJECT=IMS, you would specify DBD and DSN. - OBJECT indicates that only the specified object is to be associated with the notification. When you specify ACCESS=OBJECT, you can specify either the EDMNAME or the fully qualified data object name. For example, for OBJECT=IMS, you would specify DBD, DSN, and SEGMENT.
<i>edmtime</i>	You can specify a particular registered source segment, record, or table by using its EDMNAME. This variable supports delimited strings, but you must enclose them in quotation marks.
<i>dbdname</i>	When used alone, allows you to specify the database description (DBD) name of a set of IMS segments or VSAM records. When you use the DBD name as part of a fully qualified name, this name allows you to specify a particular IMS segment or VSAM record.
<i>data_set_name</i>	Specifies the data set name of a particular IMS segment or VSAM record as part of a fully qualified name.
<i>segment_name</i>	Specifies a particular IMS segment as part of a fully qualified name. You can use this variable multiple times (up to 255) in a single statement to associate multiple segments with the notification.
<i>ssid</i>	You can specify the subsystem ID of a particular set of DB2 tables when used alone, or a particular DB2 table when used as part of a fully qualified name
<i>tbcreator</i>	Specifies the creator of a particular DB2 table as part of a fully qualified name. This variable supports delimited strings, but you must enclose them in quotation marks. Note: tbcreator cannot handle DB2 long names and is limited to 8 bytes.
<i>table_name</i>	Specifies a particular DB2 table as part of a fully qualified name. You can use this variable multiple times (up to 255) in a single statement to associate multiple tables with the notification. These tables must be in the same subsystem and have the same creator ID. This variable supports delimited strings, but you must enclose them in quotation marks. Note: table_name cannot handle DB2 long names and is limited to 18 bytes.

If the DB2 ECCR is active when you run the create-event-marker utility to update the edition level, you must refresh the ECCR. To do so, run the `MODIFY job_name,REFRESH` command (where `job_name` is the name of the MVS batch job or started task that runs the DB2 ECCR). This ensures that the DB2 ECCR reads the new edition level in the PowerExchange repository.

Note: Alternatively, you can stop and restart the DB2 ECCR with the `WARM START` keyword.

EDMXLUTL Utility Example

The following example JCL creates an event marker when the edition level changes. You can find this example in the #EDMLUTB member of the HLQ.SAMPLIB sample library (where HLQ is the high-level qualifier specified at installation).

```
//          JOB
//*-----*
//*  DETAIL Change Capture - EVENT MARKER UTILITY TO CREATE SPECIAL EVENT
//*                      RECORD TO REFLECT A CHANGE IN EDITION LEVELS
//*-----*
//*  REPLACE THE FOLLOWING ITEMS WITH PROPER INSTALLATION VALUES
//*  1. JCL DATA SET NAMES
//*  2. EDMSYSIN DD CONTROL CARD
//*-----*
//EDMUTIL  EXEC PGM=EDMXLUTL
//STEPLIB  DD DISP=SHR,DSN=HLQ.LOAD          <=== CDM LOADLIB
//EDMPARMS DD DISP=SHR,DSN=YOUR.USERLIB      <=== EDMSDIR,EDMUPARM
//EDMMMSG  DD SYSOUT=*
//EDMSYSIN DD *
EVENT TYPE=BASEEDM NOTIFY=EDITION OBJECT=DB2 ACCESS=OBJECT          X
        EDMNAME=EDM.EDMNAME1
/*
```

The following lines show the messages that result after you run the create-event-marker utility.

Sample Messages for the Create an Event Marker Utility

```
DTLEDM175015I Control card read from EDMSYSIN
*
*  Do EVENT mark for EDMNAME=VSAM.API.SOURCE
*
EVENT
        TYPE=BASEEDM
        NOTIFY=ENDCOPY
        OBJECT=IMS
        ACCESS=OBJECT
        EDMNAME=VSAM.API.SOURCE
DTLEDM175015I Executing EVENT command; command messages may follow.
        Event type=BASEEDM
DTLEDM175025I Event Mark Notify=ENDCOPY Summary:
        Event Mark Logger RBA . . . . . :C4C7D2D340400000001E466400000000
        Event Sequence number . . . . . : 0000001E466400000000
        Event Edition number. . . . . : B42B13970E162802
        Event Source EDMNAME . . . . . : VSAM.API.SOURCE
        Related Target EDMNAME . . . . . : DB2.DEAG.RDADGK.APITARGET
```

CHAPTER 17

HOSTENT - TCP/IP Address Reporter Utility

This chapter includes the following topics:

- [HOSTENT Utility Overview, 232](#)
- [Supported Operating Systems for the HOSTENT Utility , 232](#)
- [Running the HOSTENT Utility on i5/OS, 233](#)
- [Running the HOSTENT Utility on Linux and UNIX, 233](#)
- [Running the HOSTENT Utility on z/OS , 233](#)
- [HOSTENT Utility Usage Notes, 234](#)
- [HOSTENT Utility Output, 235](#)
- [HOSTENT Utility on i5/OS Example, 235](#)
- [HOSTENT Utility on Linux and UNIX Example, 236](#)
- [HOSTENT Utility on z/OS Example, 236](#)

HOSTENT Utility Overview

Use the HOSTENT utility to:

- Display the TCP/IP host name and address for a system.
- Diagnose problems with PowerExchange communication.

Supported Operating Systems for the HOSTENT Utility

The HOSTENT utility can run on the following operating systems:

- i5/OS
- Linux and UNIX
- z/OS

Running the HOSTENT Utility on i5/OS

To run the HOSTENT utility on i5/OS:

- Enter the following command:

```
CALL HOSTENT
```

Running the HOSTENT Utility on Linux and UNIX

To run the HOSTENT utility on Linux and UNIX:

- Enter the following command:

```
hostent
```

Running the HOSTENT Utility on z/OS

Use the version of the HOSTENT TCP/IP Address Reporter utility for your TCP/IP environment.

The following table lists the HOSTENT versions by type of TCP/IP environment:

HOSTENT Version	Environment
HOSTENT	Standard z/OS Communications Server
HOSTENT2	Computer Associates CA-TCPAccess Communications Server
HOSTENT3	Native MVS Sockets

You can run the HOSTENT utility from the TSO/E command line or by submitting a z/OS job.

To run the HOSTENT utility from the command line, use the following statement:

```
call 'hlq.LOADLIB(HOSTENT) '
```

Use the sample JCL in the HOSTENT member of the RUNLIB library to create a job to run the utility. The sample JCL contains the following statements:

```
//STEP1 EXEC PGM=HOSTENT
//STEPLIB DD DSN=&SCERUN,DISP=SHR
// DD DSN=&HLQ..LOADLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//
```

JOB

Initiates the job.

EXEC PGM=HOSTENT

Invokes the utility.

STEPLIB DD

Defines the PowerExchange LOAD library that contains the utility.

SYSPRINT DD

Defines the print location for the report.

HOSTENT Utility Usage Notes

Consider the following points before using the HOSTENT utility:

- PowerExchange uses the TCP/IP resolver to translate the host name of the TCP/IP stack into an IP address.
- On z/OS and OS/390 the resolver queries the local host table. On i5/OS, Linux, and UNIX, the resolver queries the name server before it queries the local host table.
- On i5/OS, z/OS, and OS/390, PowerExchange uses the primary interface address of the TCP/IP stack to verify the licence if the resolver cannot find the host name.
- Operating systems can run more than one TCP/IP stack. Ensure that the HOSTENT runs on the TCP/IP stack that is used by PowerExchange. You cannot specify a stack name in the HOSTENT parameters.

HOSTENT Utility Resolver Details

The resolver uses the local site tables to look up the official host name and address. The resolver does not use name servers.

For z/OS 1.2 or later, you can add the following DD statement to the HOSTENT JCL to get a resolver trace to assist in diagnosis:

```
//SYSTCPT DD SYSOUT=*
```

This reports the configuration data sets and methods of look-up that the resolver uses.

HOSTENT Utility Output

The following table describes the output messages generated by HOSTENT:

Operating System	Message	Description
i5/OS, Linux, UNIX, z/OS	gethostname() gives <i>host name</i>	Displays the host name of the TCP/IP stack. On z/OS and OS/390 systems, you can find the gethostname() details in the TCPIP.DATA file specified in the TCP/IP stack. On i5/OS, Linux, and UNIX, you can find the gethostname() details in the TCPIP.DATA file used by PowerExchange.
i5/OS, Linux, UNIX, z/OS	official hostname <i>host name.domain name</i>	Displays the host name returned by the resolver. The resolver looks up the given host name to find the fully qualified name including domain name. This also displays: <ul style="list-style-type: none">- Alias names found by the resolver.- TCP/IP address as returned by the resolver. PowerExchange uses this address to validate the license.
i5/OS, Linux, UNIX, z/OS	reporting on hostname <i>host name</i>	Displays the host name. The TCP/IP resolver uses the following methods to find the host name: <ul style="list-style-type: none">- Looks up the host name from a local hosts file.- Uses the gethostbyname() system call to look up host names from a name server. PowerExchange uses these details to validate the license.
i5/OS, z/OS	gethostid() gives: <i>nnn.nnn.nnn.nnn</i>	Displays the primary interface address of the TCP/IP stack. If the TCP/IP resolver cannot find the host name, PowerExchange uses this address to validate the license. On z/OS or OS/390 systems, the gethostid() details are specified in the TCP/IP stack in the PRIMARYINTERFACEADDRESS parameter of the PROFILE data set.
z/OS	resolver gives domainname: <i>domain name</i>	Displays the domain name as determined by the local resolver configuration data set. PowerExchange does not use this address to validate the license.
z/OS	resolver gives hostname : <i>host name</i>	Displays the host name as determined by the local resolver configuration data set. PowerExchange does not use this address to validate the license.

HOSTENT Utility on i5/OS Example

The following command displays the TCP/IP host address and host name of the system on which it was run:

```
CALL HOSTENT
```

The resulting output is:

```
gethostid() gives: nnn.nnn.nnn.nnn
gethostname() gives host name
reporting on hostname host name
official hostname: host name
address: nnn.nnn.nnn.nnn
```

HOSTENT Utility on Linux and UNIX Example

The following command displays the TCP/IP host address and host name of the system on which it was run:

```
hostent
```

The resulting output is:

```
gethostname() gives host name
reporting on hostname host name
official hostname: host name
address: nnn.nnn.nnn.nnn
```

HOSTENT Utility on z/OS Example

The following statement displays the TCP/IP host address and host name of the system on which it was run:

```
//STEP1 EXEC PGM=HOSTENT,
//          PARM='/'
//STEPLIB DD DSN=&SCERUN,DISP=SHR
//          DD DSN=&HLQ..LOADLIB,DISP=SHR
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
```

The resulting output is:

```
HOSTENT:
gethostid() gives: nnn.nnn.nnn.nnn
resolver gives hostname : host name
resolver gives domainname: domain name
gethostname() gives host name
reporting on hostname host name
official hostname: host name.domain name
alias: host name
address: nnn.nnn.nnn.nnn
```

CHAPTER 18

PWXCATMY - MySQL Catalog Utility

This chapter includes the following topics:

- [PWXCATMY Utility Overview, 237](#)
- [Supported Operating Systems for the PWXCATMY Utility, 238](#)
- [PWXCATMY Operation Types, 238](#)
- [Command Syntax and Parameters, 239](#)

PWXCATMY Utility Overview

Use the PWXCATMY utility to create, prepare, and manage the PowerExchange catalog tables that are required for PowerExchange CDC for MySQL sources. The utility can also report information from the catalog tables.

The catalog tables store MySQL source table definitions. When PowerExchange receives DDL change events of CDC interest for the source tables, PowerExchange updates the source table definitions in the catalog to avoid errors when reading change data for the DDL-updated tables. Typically, these DDL changes are those that require you to re-create or modify the capture registrations and extraction maps for the source tables, for example, column add, drop, or rename operations and table drop or rename operations.

With this utility, you can perform the following tasks:

- Create the catalog tables, PWXCatTables and PWXCatUpdates, in the MySQL source database or in another local or remote MySQL database.
- Show the DDL statements for creating the catalog tables.
- Verify that the catalog tables exist and have the correct format.
- Create point-in-time snapshots of the source table definitions and write this information to the catalog. You must take an initial snapshot of all tables from which change data is captured prior to starting change data capture. In the utility, this task is called a REGISTER operation.
- Remove source table definitions from the catalog. In the utility, this task is called a UNREGISTER operation.
- List the source table names for which table definitions exist in the catalog.
- Dump the source table definitions that are currently stored in the catalog.
- Drop the catalog tables.

For more information about the catalog, see the *PowerExchange CDC Guide for Linux, UNIX, and Windows*.

Supported Operating Systems for the PWXCATMY Utility

The PWXCATMY utility runs on supported Linux or Windows systems.

For more information about supported operating systems, see the *PowerExchange Installation and Upgrade Guide*.

PWXCATMY Operation Types

Each PWXCATMY command must include the OPERATION parameter. This parameter indicates the type of operation that the utility will perform.

The PWXCATMY utility can perform the following types of operations:

CREATE

Creates the catalog tables in a MySQL database on the catalog host machine. The catalog can be in the MySQL source database or in another local or remote MySQL database, which is on a Linux or Windows machine.

DROP

Drops the existing catalog tables.

DUMP

Dumps the last table definition that is recorded in the catalog. You can select the tables for which to dump table definitions based on a schema name, specific table names, table-name masks, or an instance name that is in a PowerExchange registration group definition for a MySQL source. Alternatively, you can specify a binary log coordinate to indicate a point in binary log for the DUMP operation. In this case, the DUMP operation displays the most recent MySQL table definition prior to the coordinate position.

LIST

Lists the source table names associated with the table definitions in the catalog, for either all MySQL servers or a source server that is accessed with the source connection parameters you specify.

REGISTER

Takes an initial point-in-time snapshot of the MySQL source table definitions and records the snapshot definitions in the catalog tables. You can select the tables for which to take a snapshot based on a schema name, specific table names, table-name masks, or the instance name from the PowerExchange registration group definition for the MySQL source. The easiest way to snapshot all tables that are registered for CDC under a registration group is to enter the instance name.

SHOWDDL

Displays the DDL statements for creating the catalog tables.

UNREGISTER

Removes source table definitions from the catalog. You can select the source tables for which to remove the definitions.

VERIFY

Verifies that the catalog tables exist and have the correct format.

Command Syntax and Parameters

The parameters that you can enter with the PWXCATMY command vary by operation type.

Tip: To get help information about a command for any operation type, use the following command:

```
PWXCATMY HELP=operation_type
```

Where *operation_type* is CREATE, DROP, DUMP, LIST, REGISTER, SHOWDDL, UNREGISTER, or VERIFY.

Command Syntax by Operation Type

Use the following syntax to issue a PWXCATMY command for a specific operation type. In the syntax, square brackets [] indicate optional parameters. After the initial PWXCATMY command, you can enter the parameters in any order.

For a CREATE operation:

```
PWXCATMY OPERATION=CREATE catalog_connection_parameters
```

For a DROP operation:

```
PWXCATMY OPERATION=DROP catalog_connection_parameters
```

For a DUMP operation:

```
PWXCATMY OPERATION=DUMP source_connection_parameters catalog_connection_parameters  
table_parameters [COORD=coordinate_parameter]
```

For a LIST operation:

```
PWXCATMY OPERATION=LIST [source_connection_parameters] catalog_connection_parameters
```

For a REGISTER operation:

```
PWXCATMY OPERATION=REGISTER source_connection_parameters catalog_connection_parameters  
table_parameters
```

For a SHOWDDL operation:

```
PWXCATMY OPERATION=SHOWDDL
```

For a UNREGISTER operation:

```
PWXCATMY OPERATION=UNREGISTER source_connection_parameters catalog_connection_parameters  
table_parameters
```

For a VERIFY operation:

```
PWXCATMY OPERATION=VERIFY catalog_connection_parameters
```

Command Parameters

catalog_connection_parameters

CATHOSTNAME={*catalog_host_name*||localhost**}**

Optional. The host name or IP address of the system that contains the catalog tables for MySQL sources, or the value **localhost** if the catalog is on the system where PWXCATMY runs. Default is **localhost**.

To use a port number other than the default port, append the port number in the following format:
host_name,port_number.

CATUSERNAME=*catalog_host_username*

A user name that has the privileges required to connect to the catalog host during a PWXCATMY operation. This user must have CREATE, DROP, SELECT, UPDATE, and DELETE privileges on the catalog tables.

CATPASSWORD=*catalog_host_password*

The password that is associated with the specified catalog user.

CATSCHEMA=*catalog_table_schema*

The name of the schema for the catalog tables.

COORD=*binlog_coordinate*

A binary log coordinate that you can optionally use to mark a point in binary log for a DUMP operation. With this parameter, the DUMP operation displays the most recent source table definition prior to this coordinate position. Without this parameter, the DUMP operation displays the last table definition in the catalog.

source_connection_parameters

HOSTNAME=*{source_host_name|localhost}*

Optional. The host name or IP address of the MySQL source system.

To use a port number other than the default port, append the port number in the following format:
host_name,port_number.

USERNAME=*source_host_username*

Required. A user name that has the authority to connect to the source. This user must have at least the SELECT privilege on the source tables.

PASSWORD=*source_host_password*

Optional. The password that is associated with the specified user.

table_parameters

Enter at least one of the following parameters to identify the source tables to register or dump:

SCHEMA=*source_schema_name*

The schema name to use by default for any source table names that are specified without a schema for REGISTER or DUMP operations. You can enter this parameter only once.

TABLE=*source_table*

A MySQL source table name. You can enter this parameter multiple times, each for a different source table.

MATCH=*source_table_mask*

A pattern that matches one or more source table names. The mask specifies part of table name and uses the MySQL percentage (%) and underscore (_) wildcard characters to represent the remaining portion or portions. The percentage (%) character represents zero or more characters, and the (_) score represents a single character only. You can enter this parameter multiple times, each for a different set of source tables.

INSTANCE=*source_instance*

An instance name that appears in a PowerExchange registration group definition for a MySQL source in the PowerExchange Navigator. The easiest way to snapshot all tables that are registered for CDC under a registration group is to enter the instance name.

CHAPTER 19

PWXUCCLPRT - Print Log Summary Utility

This chapter includes the following topics:

- [PWXUCCLPRT Utility Overview, 241](#)
- [Supported Operating Systems for the PWXUCCLPRT Utility, 241](#)
- [Command Syntax for the PWXUCCLPRT Utility, 242](#)
- [PWXUCCLPRT Utility Parameters, 242](#)
- [Example PWXUCCLPRT Parameter File , 244](#)
- [Examples of PWXUCCLPRT Output, 244](#)

PWXUCCLPRT Utility Overview

PWXUCCLPRT reads PowerExchange Logger for Linux, UNIX, and Windows logs and prints a summary of the log content to the PowerExchange log.

Optionally, you can use this utility to perform the following tasks:

- Write the log header or commit restart tokens to a comma delimited file.
- Write the records to a file in hexadecimal format.

For more information about the PowerExchange Logger, see the *PowerExchange CDC Guide for Linux, UNIX, and Windows*.

Supported Operating Systems for the PWXUCCLPRT Utility

The PWXUCCLPRT utility runs on computers that have the following types of operating systems:

- Linux
- UNIX
- Windows

For more information about supported operating systems, see the *PowerExchange Installation and Upgrade Guide*.

Command Syntax for the PWXUCCLPRT Utility

Use the following syntax:

```
pwxucclprt [cs=<parameter_file>] [dbid=<collection_id>] [file=<condense_file>]
```

From the command line, supply a parameter file, collection ID, or file. You must supply at least one of these parameters.

The following table describes the command parameters:

Parameter	Description
cs	You can use the cs parameter to point to a parameter file that contains the parameters you need. For example, <code>pwxucclprt "cs=parameters.txt"</code>
dbid	Specifies the collection ID. For example, <code>pwxucclprt dbid=ORCL</code>
file	Specifies the condense file. For example, <code>pwxucclprt file="orcl.CND.CP191031.T0819001"</code>

PWXUCCLPRT Utility Parameters

The parameter file specifies an INPUT statement to filter the input and an OUTPUT statement to specify the output format. This section describes the parameters for the INPUT and OUTPUT statements in the parameter file.

Note: You must specify a DBID parameter in the INPUT statement or a FILE parameter in the OUTPUT statement. All other parameters are optional.

PWXUCCLPRT INPUT Statement Parameters

The INPUT statement has the following parameters:

DBID=collection_id

Specifies the collection identifier.

STARTIME=YYYYMMDDHHMISS

Filters out all events before the specified time.

ENDTIME=YYYYMMDDHHMISS

Filters out all events after the specified time.

Note: Time filters are inclusive. For example, `STARTIME=201608 ENDTIME=201608` selects all of the data that was generated in August 2016.

STARTSEQUENCE=*startsequence*

The start sequence value in hexadecimal format. Filters out all events before the specified sequence. PWXUCCLPRT does not check the sequence length.

ENDSEQUENCE=*endsequence*

The end sequence value in hexadecimal format. Filters out all events after the specified sequence. For example, in Oracle Capture: `STARTSEQUENCE=D40000023CE27D ENDSEQUENCE=D40000023CE300` selects all of the data between commit SCN 37544573 and 37544704.

Sequence filters are inclusive. The utility does not check the sequence length.

TAGLIST=(*tag, tag, ...*)

Specifies a list of capture tags to filter the data.

Note: In the case of flexible condense, the utility processes the first group by default. To process a different group, you must supply a tag from the group you want to process. To process all tags from the group, supply an asterisk(*) in the tag list. For example: (ORAORCLdept1,*)

UIDLIST=(*id, id, ...*)

Specifies a list of user IDs to filter the data. The filter is not case sensitive.

FILELIST=(*file, file, ...*)

Specifies a list of specific files to process. For example: ("/ccl/group1/grp1.CND.CP160802.T0916001", "/ccl/group1/grp1.CND.CP160802.T0917001")

RECORDLIMIT=*n*

Limits the number of records that the utility processes after the filter criteria are applied.

ENCRYPTOPT={*AES128|AES192|AES256*}

The AES encryption algorithm used for encrypting PowerExchange log files. By default, encrypted condense records are dumped in encrypted format. To dump the data in unencrypted format, you must specify the encryption options used by pwxcl.

ENCRYPTPWD=*encrypted_encryption_password*

The password used for encrypting PowerExchange log files. By default, encrypted condense records are dumped in encrypted format. To dump the data in unencrypted format, you must specify the encryption options used by pwxcl.

PWXUCCLPRT OUTPUT Statement Parameters

The OUTPUT statement has the following parameters:

FORMAT={*SUMMARY|TOKENS|CSV|DUMP*}

Specifies the output format. Specify one of the following options:

SUMMARY

The default. Write record counts to the log.

TOKENS

Write the commit tokens to the output file in CSV format.

CSV

Write the PowerExchange Logger for Linux, UNIX, and Windows file header information to the output file in CSV format.

DUMP

Write the PowerExchange Logger for Linux, UNIX, and Windows record information to the output file in hexadecimal format.

FILE=*file_name*

Specifies the output file name to use for all format options other than SUMMARY. Default is pwxucclprt.csv.

Example PWXUCCLPRT Parameter File

The following parameter file example shows the INPUT and OUTPUT statement specifications. The output will be written to the `PWXUCCLPRT DUMP.txt` file.

```
INPUT
DBID=ORCL
STARTTIME=201608141729
ENDTIME=201608141730
STARTSEQUENCE=D40000023CE27D00000000000000000023CE27C0000000000000145000007FF001000020000
ENDSEQUENCE=D40000023CE2880000000000000FFFFFFFFFFFFFFFFF00000145000008330010FFFF0000
RECORDLIMIT=100
TAGLIST=(ORAORCLdept1,*)
UIDLIST=(SYS)
FILELIST=("/ccl/group1/grp1.CND.CP160802.T0916001","/ccl/group1/
grp1.CND.CP160802.T0917001")
ENCRYPTOPT=AES256
ENCRYPTPWD=A1B1C1D1E1F1G1
;
OUTPUT
FORMAT=dump
FILE="PWXUCLPRT_DUMP.txt"
;
```

Examples of PWXUCCLPRT Output

The following examples show the different types of output formats that the PWXUCCLPRT utility can produce.

Example 1. CSV Output Format

The following example shows the PWXUCCLPRT output in CSV format.

Parameter File Input

The example uses the following parameter file input:

```
INPUT
DBID=ORCL
OUTPUT
FORMAT=CSV
FILE=C:\Users\user\Documents\temp\PWXUCLPRT.CSV
```

Messages

If the command is successful, the following messages are written to the message log file.

```
PWX-36993 Processing file C:\Users\user\resources\ccl\group2\orcl.CND.CP191031.T0819001.
PWX-36994 File totals for ORAORCLdept1: Inserts 20000. Updates 0. Deletes 20000. UOWs 1. Largest UOW 40000.
PWX-36994 File totals for ORAORCLempl: Inserts 70000. Updates 0. Deletes 70000. UOWs 1. Largest UOW 140000.
PWX-36995 File totals: Inserts 90000. Updates 0. Deletes 90000. UOWs 1. Largest UOW 180000.
PWX-36996 Run totals for ORAORCLdept1: Inserts 20000. Updates 0. Deletes 20000. UOWs 1. Largest UOW 40000.
PWX-36996 Run totals for ORAORCLempl: Inserts 70000. Updates 0. Deletes 70000. UOWs 1. Largest UOW 140000.
PWX-36997 Run totals: Inserts 90000. Updates 0. Deletes 90000. UOWs 1. Largest UOW 180000.
```

Output

The utility writes the summary to a CSV file:

File#	Record#	Offset	Type	ImType	ImLen	Tag#	TagName	Sequence	Restart	LRSN	TStamp	ChgType
1910310819002	2	212	20	0	71	21091	ORAORCLempl					
D4000005F2DDBB00000000000000000005F2AE6000000000000042E900000F33001000020000								000005F2AE605947602939D214AD			D6F40C1F6B700000	
20191031081700000000	D		SYS									
1910310819002	3	404	20	0	74	21091	ORAORCLempl					
D4000005F2DDBB00000000000000000005F2AE6000000001000042E900000F3400D400010000								000005F2AE605947602939D214AD			D6F40C1F6B700000	
20191031081700000000	D		SYS									
1910310819002	4	587	20	0	73	21091	ORAORCLempl					
D4000005F2DDBB00000000000000000005F2AE6000000002000042E900000F35002C00010000								000005F2AE605947602939D214AD			D6F40C1F6B700000	
20191031081700000000	D		SYS									
1910310819002	5	769	20	0	73	21091	ORAORCLempl					
D4000005F2DDBB00000000000000000005F2AE6000000003000042E900000F35017000010000								000005F2AE605947602939D214AD			D6F40C1F6B700000	
20191031081700000000	D		SYS									
1910310819002	6	951	20	0	75	21091	ORAORCLempl					
D4000005F2DDBB00000000000000000005F2AE6000000004000042E900000F3600C400010000								000005F2AE605947602939D214AD			D6F40C1F6B700000	
20191031081700000000	D		SYS									
1910310819002	7	1135	20	0	73	21091	ORAORCLempl					
D4000005F2DDBB00000000000000000005F2AE6000000005000042E900000F37001C00010000								000005F2AE605947602939D214AD			D6F40C1F6B700000	
20191031081700000000	D		SYS									
1910310819002	8	1317	20	0	73	21091	ORAORCLempl					
D4000005F2DDBB00000000000000000005F2AE6000000006000042E900000F37016000010000								000005F2AE605947602939D214AD			D6F40C1F6B700000	
20191031081700000000	D		SYS									

Example 2. Dump Output Format

The following example shows the PWXUCCLPRT output in dump format.

Parameter File Input

The example uses the following parameter file input:

```
INPUT
DBID=ORCL
RECORDLIMIT=5
OUTPUT
FORMAT=DUMP
FILE=C:\Users\user\Documents\temp\PWXUCCLPRT.TXT
```

Messages

If the command is successful, the following messages are written to the message log file.

```
PWX-36993 Processing file C:\Users\user\resources\ccl\group2\orcl.CND.CP191031.T0819001.
PWX-36994 File totals for ORAORCLempl: Inserts 0. Updates 0. Deletes 5. UOWs 0. Largest UOW 0.
PWX-36995 File totals: Inserts 0. Updates 0. Deletes 5. UOWs 0. Largest UOW 0.
PWX-36996 Run totals for ORAORCLempl: Inserts 0. Updates 0. Deletes 5. UOWs 0. Largest UOW 0.
PWX-36997 Run totals: Inserts 0. Updates 0. Deletes 5. UOWs 0. Largest UOW 0.
```

Output

The utility writes the output in dump format:

```
Dumping record#2, Offset 0x000000D4, Tag ORAORCLempl

0x0000 00C91C00 05000000 534D4954 48000500 *.....SMITH...*
0x0010 0000434C 45524B00 DE1E0031 39383031 *..CLERK....19801*
0x0020 32313730 30303030 30303030 30303000 *217000000000000.*
0x0030 38303000 00000000 0000FF00 00000000 *800.....*
```


CHAPTER 20

PWXUCDCT - Logger for Linux, UNIX, and Windows Utility

This chapter includes the following topics:

- [PWXUCDCT Utility Overview, 247](#)
- [Supported Operating Systems for the PWXUCDCT Utility, 248](#)
- [Control Statement Syntax for PWXUCDCT Commands, 248](#)
- [PWXUCDCT Commands and Parameters, 248](#)
- [Running the PWXUCDCT Utility, 254](#)
- [Usage Notes for the PWXUCDCT Utility, 255](#)
- [Examples of PWXUCDCT Utility Commands, 255](#)

PWXUCDCT Utility Overview

Use the PWXUCDCT utility to manage files and print reports for the PowerExchange Logger for Linux, UNIX, and Windows.

With the utility, you can perform the following tasks:

- Manually back up the CDCT file if the backups that are automatically generated at PowerExchange Logger initialization and termination are not available or not recent.
- Derive a CDCT file backup that can be used for a restore operation based on the PowerExchange Logger log files.
- Restore the CDCT file from a backup.
- Delete expired CDCT records and any PowerExchange Logger log files associated with those records.
- Delete orphaned PowerExchange Logger log files that are not referenced by any CDCT record.
- Print reports on PowerExchange Logger pwxcl configuration parameters, the CDCT file contents, and current, orphaned, and expired log files.

For more information about the PowerExchange Logger, see the *PowerExchange CDC Guide for Linux, UNIX, and Windows*.

Supported Operating Systems for the PWXUCDCT Utility

The PWXUCDCT utility runs on computers that have the following types of operating systems:

- Linux
- UNIX
- Windows

For more information about supported operating systems, see the *PowerExchange Installation and Upgrade Guide*.

Control Statement Syntax for PWXUCDCT Commands

Use the following general syntax to specify control statements for the PWXUCDCT utility:

```
PWXUCDCT
  CMD=command_name
  [CONFIG=override_dbmover.cfg]
  [CS=override_pwxocl.cfg]
  [LICENSE=override_license.key]
  command-specific parameters
```

The following syntax rules apply:

- You can specify the optional CONFIG, CS, and LICENSE parameters on any command. Informatica recommends that you specify the CS parameter because each command must run under a specific PowerExchange Logger configuration for a database instance. If you do not specify a PowerExchange Logger configuration file, the PowerExchange Logger uses the pwxocl file in the top-level installation directory.
- Other parameters are specific to the command that is being issued. You can enter these command-specific parameters in any order.
- You cannot define the parameters in a separate file and then reference that file in the command syntax.

RELATED TOPICS:

- [“PWXUCDCT Commands and Parameters” on page 248](#)
- [“Usage Notes for the PWXUCDCT Utility” on page 255](#)
- [“Examples of PWXUCDCT Utility Commands” on page 255](#)

PWXUCDCT Commands and Parameters

This section describes the commands that you can enter in the CMD statement of the PWXUCDCT syntax and the command-specific parameters.

It also describes the CONFIG, CS, LICENSE parameters that you specify for any command.

Commands

This topic summarizes the commands that you can issue to the PWXUCDCT utility, including any command-specific parameters.

The following table describes each command:

Command	Description	Command-specific Parameters ¹
CONVERT_CDCT	<p>If you upgrade to 9.5.1 HotFix 1 or later from an earlier release, you can issue this command to manually perform a one-time conversion of the CDCT file to the new format. Alternatively, the first time the PowerExchange Logger is warm started, it automatically converts the CDCT file to the new format.</p> <p>The conversion creates a CDCT_<i>dbid</i> file instance from the original CDCT file. Ensure that the <i>dbid</i> value in the CDCT file name matches the DBID parameter value in the PowerExchange Logger pwxcl configuration file under which you run the command.</p> <p>Note: If the old CDCT file contains information for multiple database instances, you must run this command multiple times, once for each instance. Each time you run the command, ensure that the CS parameter points to the correct pwxcl configuration file for the instance.</p>	None
CREATE_CDCT_BACKUP	<p>Manually creates a backup of all records in a CDCT file instance for a source database based on the latest configuration incarnation.</p> <p>The <i>dbid</i> value in the CDCT file name must match the DBID parameter value in the pwxcl configuration file.</p> <p>Note: The PowerExchange Logger automatically generates a backup at initialization and at termination.</p>	BACKUPFILE
DELETE_EXPIRED_CDCT	<p>This command is deprecated but is still supported for backward compatibility. Use DELETE_EXPIRED_FILES instead.</p>	None
DELETE_EXPIRED_FILES	<p>Deletes the log files for which the retention period has expired and the CDCT records that reference those expired logs. For this command to work, you must set the LOGGER_DELETES_EXPIRED_CDCT_RECORDS parameter to N in the pwxcl configuration file.</p> <p>If you set the LOGGER_DELETES_EXPIRED_CDCT_RECORDS parameter to Y, or if you do not specify this parameter, the command does not work.</p>	None
DELETE_ORPHAN_FILES	<p>Deletes PowerExchange Logger log files that are not referenced by any record in the CDCT file.</p>	None

Command	Description	Command-specific Parameters ¹
DERIVE_CDCT_BACKUP	<p>If the CDCT file is corrupted or deleted, and if a recent CDCT backup is not available or the latest available backup would result in significant reprocessing of data, use this command to derive a backup file for recovery purposes.</p> <p>The command uses the EXTERNAL_CAPTURE_MASK parameter value from the PowerExchange Logger configuration file or the <i>external_capture_mask</i> positional parameter from the group definition file to generate a list of PowerExchange Logger log files. The command then uses the content of these log files to generate a text file that can be used as input to the RESTORE CDCT command.</p> <p>Do not use this command if the PowerExchange Logger log files were also corrupted or deleted.</p> <p>Tip: Use the PREVBACKUPFILE parameter to supply the name of the last available backup file. By using a previous backup file, you preserve more historic information in the CDCT file. The utility will add any log files that were created since the backup was taken to the derived backup file.</p>	BACKUPFILE [PREVBACKUPFILE]

Command	Description	Command-specific Parameters ¹
REPORT_CDCT	<p>Prints the contents of the CDCT file. This information is primarily for debugging purposes.</p> <p>For the current Logger configuration incarnation, the report shows:</p> <ul style="list-style-type: none"> - Incarnation identifier, status, and reason (Rsn) for creation. The reason can be a cold start or a change in the configuration. - Source instance (or DBID) name and image type. - Number of groups defined in the group definition file. If no groups are defined, the default of 1 is used. - Type of AES encryption algorithm, if log file encryption is enabled. - Begin and end restart and sequence tokens. <p>For each Logger group, the report shows:</p> <ul style="list-style-type: none"> - Group number and name. - Incarnation to which the group belongs. - Path to the group log files. - Registration count. - Log file count, and the first and current log sequence numbers. - Status - Oldest log file timestamp. <p>For each registration, the report shows:</p> <ul style="list-style-type: none"> - Registration tag name and status. - Incarnation and group to which the registration belongs. - Activation date and inactivation date, if available. - Default schema name. <p>For each PowerExchange Logger log file, the report shows:</p> <ul style="list-style-type: none"> - Log file path and file name, and sequence number. - Configuration incarnation and group to which the log file belongs. - File open and close timestamps. - Record count, commit count, and whether the log file contains uncommitted data. - Whether the log file is encrypted, and a value used to test the encryption key. - Status and Fmt version. - Begin and end restart tokens. 	[report_file_name]
REPORT_CDCT_FILES	<p>Reports the following information for each PowerExchange Logger log file that is recorded in the CDCT file:</p> <ul style="list-style-type: none"> - Log file path and file name, and sequence number. - Configuration incarnation and group to which the log file belongs. - File open and close timestamps. - Record count, commit count, and whether the log file contains uncommitted data. - Whether the log file is encrypted, and a value used to test the encryption key. - Status and Fmt version. - Begin and end restart tokens. <p>This information is the same that reported in the CCL Files section of the REPORT_CDCT report.</p>	[report_file_name]

Command	Description	Command-specific Parameters ¹
REPORT_CONFIG	Lists the parameter settings that are defined in the associated PowerExchange Logger pwxcl configuration file. If you created a group definition file and specified it in the GROUPDEFS parameter in the pwxcl file, the command also reports the group statements in the group definition file.	[report_file_name]
REPORT_EXPIRED_CDCT	This command is deprecated but is still supported for backward compatibility. Use REPORT_EXPIRED_FILES instead.	[report_file_name]
REPORT_EXPIRED_FILES	Lists the PowerExchange Logger log files for which the retention period has elapsed.	[report_file_name]
REPORT_FILES_BY_NAME	Lists PowerExchange Logger log files by file name. This information is based on directory information for the log files and not on the CDCT file. For each file, the command reports the following information: - Date and time when the file was written. - Sequence number - Path and file name. Also, the command reports the number of log files that match the default mask that is specified in the EXT_CAPT_MASK parameter of the pwxcl configuration file. If you specified a group definition file in the GROUPDEFS parameter of the pwxcl file, the command reports the number of log files that match any masks in the group definition file. Note: The PowerExchange Logger generates log file names that include the EXT_CAPT_MASK value, the date and time, and a sequential number. For example: MYMASK.CND.CP090813.T1748013, where 090813 is MMDDYY, 1748 is HHMM, and 013 is the generated sequential number.	[report_file_name]
REPORT_FILES_BY_TIME	Lists PowerExchange Logger log files in the order in which they were created, from earliest to latest. This information is based on directory information for the log files and not on the CDCT file. For each file, the command reports the following information: - Date and time when the file was written. - Sequence number of the file. - Path and file name. Also, the command reports the number of log files that match the default mask that is specified in the EXT_CAPT_MASK parameter of the pwxcl configuration file. If you specified a group definition file in the GROUPDEFS parameter of the pwxcl file, the command also reports the number of log files that match any masks in the group definition file.	[report_file_name]
REPORT_ORPHAN_FILES	Lists PowerExchange Logger log files that are not referenced by any record in the CDCT file.	[report_file_name]

Command	Description	Command-specific Parameters ¹
RESTORE_CDCT	Restores the CDCT file from a backup, up to a specific point in time. The PowerExchange Logger will reprocess any data that is later than this point in time. After the restore operation completes, run the DELETE_ORPHAN_FILES command.	BACKUPFILENAME [ENCRYPTEPWD] [PROGRESSFREQUENCY]
1. Optional parameters are enclosed in square brackets.		

Also, you can specify the following global parameters on any PWXUCDCT utility command: CONFIG, CS, and LICENSE. Informatica recommends that you include the CS parameter in each command statement because each command must run under a specific PowerExchange Logger configuration. Otherwise, the utility uses the default PowerExchange Logger configuration file in the installation directory.

Parameter Descriptions

The following global and command-specific parameters are used with PWXUCDCT utility commands.

BACKUPFILE=*path\file_name.txt*

Specifies the full path and file name for a CDCT backup file that you are creating or using for a restore operation. The backup file is a delimited text file.

Required in the following commands: CREATE_CDCT_BACKUP, DERIVE_CDCT_BACKUP, and RESTORE_CDCT.

CONFIG=*path\file_name*

If you specified the CONFIG parameter in the pwxcl statement when starting the PowerExchange Logger process, specify the same parameter value in the PWXUCDCT utility command for that process. The parameter specifies the full path and file name for a DBMOVER configuration file that overrides the default dbmover configuration file in the installation directory. The full path is required only if the override file does not reside in the default location. The override file takes precedence over any other override configuration file that you optionally specify with the PWX_CONFIG environment variable.

For example, you might use an override DBMOVER configuration file to split PowerExchange Logger processing across multiple database instances but maintain a separate CDCT file for each instance.

Optional in any PWXUCDCT utility command.

CS=*path\file_name*

If you specified the CS parameter in the pwxcl statement when starting the PowerExchange Logger process, specify the same parameter value in the PWXUCDCT utility command for that process. The parameter specifies the full path and file name of the PowerExchange Logger configuration file. If you specify either the CONFIG or LICENSE parameter, the CS parameter is required. You can use the CS parameter to specify a PowerExchange Logger configuration file that overrides the default pwxcl configuration file in the installation directory. The full path is required only if the override file does not reside in the default location.

Recommended in all PWXUCDCT utility commands.

ENCRYPTEPWD=*encrypted_encryption_password*

If you cold started the PowerExchange Logger from the command line with a pwxcl command that included the encryptpwd parameter, you must specify that same parameter value in the ENCRYPTEPWD

parameter in the RESTORE_CDCT command. The parameter specifies an encryption password, in encrypted format, that enables the encryption of PowerExchange Logger log files. With this password, the command can restore the CDCT file, including the encryption password that is stored in the file in encrypted format.

Tip: After you run the RESTORE_CDCT command, perform a CAPX database row test from the PowerExchange Navigator to verify that the encryption password has been successfully restored.

LICENSE=*path\file_name*

If you specified the LICENSE parameter in the pwxcl command when starting the PowerExchange Logger process, specify the same parameter value in the PWXUCDCT utility command for that process. The parameter specifies the full path and file name for a license key file that overrides the default license.key file in the installation directory. The full path is required only if the override file does not reside in the default location. The override file takes precedence over any other override license key file that you optionally specify with the PWX_LICENSE environment variable.

Optional in all PWXUCDCT commands.

PREVBACKUPFILE=*file_name*

Specifies a previous backup file to use with the DERIVE_CDCT_BACKUP command to derive a backup that is suitable for a restore operation. Use this parameter when a recent backup is not available after a failure or the latest available backup would result in significant reprocessing of data. You can use a backup file that was automatically generated at PowerExchange Logger initialization or shutdown or that you manually created. By using a previous backup file, you preserve more historic information in the CDCT file. Also, the utility will roll the CDCT contents forward based on any log files it discovers that are later than the backup point in time.

PROGRESSFREQUENCY=*number_of_records*

Specifies the frequency with which the PWXUCDCT utility displays progress information for a RESTORE_CDCT operation. The frequency is expressed as the number of records read from the CDCT backup file. Each time the utility processes this number of records, it writes progress message PWX-25132 to the console screen and PowerExchange message log. Default is to display progress information each time the utility processes approximately 1 percent of the records in the backup file.

report_file_name

Specifies a path and file name that you can specify to send report output to a file instead of to the command line screen. In the command, precede this value with a greater than (>) sign, for example:

```
>C:\Informatica\PowerExchange9.5.1\reports\expiredcdct01.txt
```

Optional in any PWXUCDCT REPORT command.

For more information about the PWXUCDCT commands, see [“Commands” on page 249](#).

Running the PWXUCDCT Utility

You can run a PWXUCDCT utility command from a command line on a Linux, UNIX, or Windows system where PowerExchange is installed.

Make sure that you run the utility under a user ID that has READ access to the PowerExchange Logger log files and control files.

To run the utility, navigate to the directory where the `pwxucdct` executable is located. By default, this directory is in the PowerExchange installation directory. Then enter `pwxucdct` followed by a command name and any relevant command parameters. Use the following syntax:

```
C:\Informatica\PowerExchangev.r.m pwxucdct [parameter1 parameter2...] cs=pwxcc1_config
CMD=command_name
```

Usage Notes for the PWXUCDCT Utility

Before you use the PWXUCDCT utility, review the following usage notes:

- You can schedule PWXUCDCT utility commands to run during off-peak hours to avoid increasing the PowerExchange Logger workload when transaction activity is high.
- If you run the PowerExchange Logger in continuous mode, do not use the `CREATE_CDCT_BACKUP` command to back up the CDCT file while it is being updated.
- By default, the PWXUCDCT utility writes output from a `REPORT` command to stdout so that you see it on screen. To send the output to a file, you must specify a *report_file_name* preceded by a greater than (>) sign, for example, `>C:\Informatica\PowerExchange9.0.0\reports\myfile.txt`. Otherwise, the PWXUCDCT utility scrolls the report lines onto the screen. For some `REPORT` commands, the utility also writes report messages to the PowerExchange message log but does not include the detailed lines that are written to stdout.
- All PWXUCDCT utility commands other than `REPORT_FILES_BY_TIME` and `REPORT_FILES_BY_NAME` reference the CDCT file.
- The CDCT file name includes the source instance name that is specified in the `DBID` parameter of the `pwxcc1` configuration file. The CDCT location is determined by the `CAPT_PATH` parameter in the `dbmover` configuration file. The location can be expressed as `$(CAPT_PATH)/CDCT_$(DBID)`.

Examples of PWXUCDCT Utility Commands

This section provides example PWXUCDCT utility commands and shows sample output where appropriate. Enter the commands from a command line.

Example 1. Creating a Backup of the CDCT File

The backups that PowerExchange automatically generated during PowerExchange Logger initialization and shutdown are not available. To create a backup of all records in the CDCT file manually, you run the `CREATE_CDCT_BACKUP` command during daily batch processing.

Enter the following command:

```
pwxucdct cmd=create_cdct_backup backupfile=C:\Informatica\PowerExchangev.r.m\backups
\backup1.txt cs=C:\Informatica\PowerExchangev.r.m\resources\pwxcc1_orcl.cfg
```

If the command is successful, the following messages are written to the message log file:

```
PWX-33314 TIMEOUTS configuration parameter is deprecated
PWX-33269 CCL configuration parameter <CHKPT_BASENAME> is deprecated.
PWX-33269 CCL configuration parameter <CHKPT_NUM> is deprecated.
PWX-25404 Processing console program. pwxucdct cmd=create_cdct_backup backupfile=C:\Informatica
\PowerExchangev.r.m\backups\backup1.txt cs=C:\Informatica\PowerExchangev.r.m\resources\pwxcc1_orcl.cfg
```

The backup1.txt file is created in the backups directory.

Tip: You can use this backup file to restore the CDCT file, if necessary. In the RESTORE_CDCT command, use the backupfile parameter to specify the backup file name and path.

Example 2. Restoring the CDCT File from a Backup File

The CDCT file has become damaged. You want to restore it from its latest backup file. You created the backup file with the CREATE_CDCT_BACKUP command based on the latest Logger configuration incarnation.

From the command line, navigate to the PowerExchange installation directory and enter the following command:

```
pxwucdct cmd=restore_cdct backupfile=C:\Informatica\PowerExchangev.r.m\backups
\backup1.txt cs=C:\Informatica\PowerExchangev.r.m\resources\pxwccl_orcl.cfg
```

If the command is successful, the following messages are written to the message log file:

```
PWX-33314 TIMEOUTS configuration parameter is deprecated
PWX-33269 CCL configuration parameter <CHKPT_BASENAME> is deprecated.
PWX-33269 CCL configuration parameter <CHKPT_NUM> is deprecated.
PWX-25404 Processing console program. pxwucdct cmd=restore_cdct backupfile=C:\Informatica\PowerExchangev.r.m
\backups\backup1.txt cs=C:\Informatica\PowerExchangev.r.m\resources\pxwccl_orcl.cfg
PWX-25200 Created CDCT file "C:\Informatica\PowerExchangev.r.m\resources\CDCT_orcl"
PWX-36937 Restore is using backup of CDCT file <C:\Informatica\PowerExchangev.r.m\resources\CDCT_orcl>
created <2013/02/21 19:03:14.000000>.
```

Example 3. Re-creating the CDCT File After a Failure

The CDCT file and all recent CDCT backup files were damaged or deleted. However, an older backup file based on a previous Logger configuration incarnation is available. To re-create the CDCT file, you first derive a backup based on the previous backup file and then restore that backup.

1. Derive a backup file from existing log files by entering the following command:

```
pxwucdct cmd=derive_cdct_backup prevbackupfile=C:\Informaticav.r.m\pxw\backups
\prev_backup0.txt backupfile=C:\Informaticav.r.m\pxw\backups\derived_backup1.txt
cs=C:\Informaticav.r.m\pxw\resources\pxwccl_orcl.cfg
```

Tip: Include the PREVBKUPFILE parameter to use a previous backup file as a starting point to recover the CDCT. If the utility discovers additional log files that are not in the previous backup, it adds them to the derived backup.

The following messages are written to the message log file:

```
PWX-33314 TIMEOUTS configuration parameter is deprecated
PWX-33269 CCL configuration parameter <CHKPT_BASENAME> is deprecated.
PWX-33269 CCL configuration parameter <CHKPT_NUM> is deprecated.
PWX-25404 Processing console program. pxwucdct cmd=derive_cdct_backup backupfile=C:\Informaticav.r.m\pxw
\backups\derived_backup1.txt cs=\Informaticav.r.m\pxw\resources\pxwccl_orcl.cfg
2 Logger file(s) found for flexible groups masks
PWX-33261 Loaded "bonus.1". Table "MYORAL.BONUS". Tag "ORAORCLbonus1"
PWX-33261 Loaded "dept.1". Table "MYORAL.DEPT". Tag "ORAORCLdept1"
PWX-33261 Loaded "emp.1". Table "MYORAL.EMP". Tag "ORAORCLempl"
PWX-33261 Loaded "o015716k.1". Table "MYORAL.ORL157_SRC_16K". Tag "ORAORCLo015716k1"
PWX-33261 Loaded "o01612k.1". Table "MYORAL.ORL161_SRC_2K". Tag "ORAORCLo01612k1"
PWX-33261 Loaded "salgrade.1". Table "MYORAL.SALGRADE". Tag "ORAORCLsalgrade1"
PWX-33263 12 registrations loaded
PWX-33264 Start of registrations for group processing
PWX-06264 Group <GROUP1> using registration <emp> reg_schema=<MYORAL> table=<EMP> with schema=<MYORAL>.
PWX-06264 Group <GROUP2> using registration <bonus> reg_schema=<MYORAL> table=<BONUS> with
schema=<MYORAL>.
PWX-06264 Group <GROUP2> using registration <dept> reg_schema=<MYORAL> table=<DEPT> with schema=<MYORAL>.
PWX-06264 Group <GROUP2> using registration <o015716k> reg_schema=<MYORAL> table=<ORL157_SRC_16K> with
schema=<MYORAL>.
PWX-06264 Group <GROUP2> using registration <o01612k> reg_schema=<MYORAL> table=<ORL161_SRC_2K> with
schema=<MYORAL>.
PWX-06264 Group <GROUP2> using registration <salgrade> reg_schema=<MYORAL> table=<SALGRADE> with
schema=<MYORAL>.
PWX-06119 Controller: added new registration tag ORAORCLbonus1
PWX-06119 Controller: added new registration tag ORAORCLdept1
PWX-06119 Controller: added new registration tag ORAORCLo015716k1
PWX-06119 Controller: added new registration tag ORAORCLsalgrade1
```


2. Restore the derived backup by entering the following command:

```
pwxcudct cmd=restore_cdct backupfile=C:\Informaticav.r.m\pwx\backups
\derived_backup1.txt cs=C:\Informaticav.r.m\pwx\resources\pwxcc1_orcl.cfg
```

The following messages are written to the message log file:

```
PWX-33314 TIMEOUTS configuration parameter is deprecated
PWX-33269 CCL configuration parameter <CHKPT_BASENAME> is deprecated.
PWX-33269 CCL configuration parameter <CHKPT_NUM> is deprecated.
PWX-25404 Processing console program. pwxcudct cmd=restore_cdct backupfile=C:\Informatica
\PowerExchangev.r.m\backups\derived_backup1.txt cs=C:\Informatica\PowerExchangev.r.m\resources
\pwxcc1_orcl.cfg
PWX-25200 Created CDCT file "C:\Informatica\PowerExchangev.r.m\resources\CDCT_orcl"
PWX-36937 Restore is using backup of CDCT file <C:\Informatica\PowerExchangev.r.m\resources\CDCT_orcl>
created <2013/02/21 19:03:14.000000>.
```

To verify that the restore operation was successful, check that the return code from the PWXUCDCT utility is zero and that messages PWX-25140 through PWX-25145 provide reasonable record counts for records read from the backup file and for records that were changed in the CDCT file.

Also, view the PWX-25132 messages that report the progress of the restore operation. PowerExchange tries to display progress information to the console approximately every 1 percent of the backup file processed. If you need to display progress information more frequently or less frequently, include the progressfrequency parameter in the restore_cdct statement to adjust the frequency.

3. Run the DELETE_ORPHAN_FILES command to delete log files that are no longer referenced by the restored CDCT file.

After you warm start the PowerExchange Logger, it re-creates the CDCT content for those files.

Example 4. Reporting and Deleting Orphan CDCT Records

You want to determine if orphaned PowerExchange Logger log files exist so that you can delete them. Orphaned log files are not referenced by any record in the CDCT file.

1. To determine if orphan log files exist, enter the following command:

```
pwxcudct cmd=report_orphan_files cs=C:\Informatica\PowerExchangev.r.m\resources
\pwxcc1_orcl.cfg
```

The following messages are displayed on screen and written to the message log file:

```
PWX-25404 Processing console program. pwxcudct cmd=report_orphan_files
REPORT FOR COMMAND REPORT_ORPHAN_FILES
PWX-25229 Started initialization of the CDCT Retention Array
PWX-25230 Retention array initialized. Files 2. CDCTs read 0. Allocated 0. Memory 0
5 Logger file(s) found for mask C:\Informatica\PowerExchange9.0.0\capture\condense0.CND.*
Total files found for masks 5
Date Time Seq File
-----
091109 1447 007 C:\Informatica\PowerExchangev.r.m\capture\condense0.CND.CP091109.T1447007
091109 1615 008 C:\Informatica\PowerExchangev.r.m\capture\condense0.CND.CP091109.T1615008
091109 1615 009 C:\Informatica\PowerExchangev.r.m\capture\condense0.CND.CP091109.T1615009
```

These messages indicate that three orphan log files exist.

2. To delete all orphan log files, enter the following command:

```
pwxcudct cmd=delete_orphan_files cs=C:\Informatica\PowerExchangev.r.m\resources
\pwxcc1_orcl.cfg
```

The following messages are displayed on screen and written to the message log file:

```
PWX-25404 Processing console program. pwxcudct cmd=delete_orphan_files
REPORT FOR COMMAND DELETE_ORPHAN_FILES
PWX-25229 Started initialization of the CDCT Retention Array
PWX-25230 Retention array initialized. Files 2. CDCTs read 0. Allocated 0. Memory 0
5 Logger file(s) found for mask C:\Informatica\PowerExchange9.0.0\capture\condense0.CND.*
Total files found for masks 5
Date Time Seq File
-----
PWX-25163 Deleted orphan file C:\Informatica\PowerExchangev.r.m\capture\condense0.CND.CP091109.T1447007
PWX-25163 Deleted orphan file C:\Informatica\PowerExchangev.r.m\capture\condense0.CND.CP091109.T1615008
PWX-25163 Deleted orphan file C:\Informatica\PowerExchangev.r.m\capture\condense0.CND.CP091109.T1615009
PWX-25162 Files not referenced in CDCT (orphans) 3
```

These messages indicate that the orphan log files were successfully deleted.

- ```
pwxucdct cmd=report_files_by_time
```

```

PXW-25404 Processing console program. pxwucdct cmd=report_files_by_time
REPORT FOR COMMAND REPORT_FILES_BY_TIME
PXW-25229 Started initialization of the CDCT Retention Array
PXW-25230 Retention array initialized. Files 2. CDCTs read 0. Allocated 0. Memory 0
2 Logger file(s) found for mask C:\Informatica\PowerExchangev.r.m\capture\condenseO.CND.*
Date Time Seq File

091109 1443 006 C:\Informatica\PowerExchangev.r.m\capture\condenseO.CND.CP091109.T1443006
091109 1615 010 C:\Informatica\PowerExchangev.r.m\capture\condenseO.CND.CP091109.T1615010

```

## Example 5. Reporting and Deleting Expired CDCT Records

**Note:** To use the DELETE\_EXPIRED\_CDCT command, you must set the `LOGGER_DELETES_EXPIRED_CDCT_RECORDS` parameter to Y in the `pxwccf` configuration file. If this parameter is set to Y or is not specified, the PowerExchange Logger does not delete the expired CDCT records until a file switch occurs.

- ```
pwxucdct cmd=report_expired_cdct cs=C:\Informatica\PowerExchangev.r.m\resources
\pwxcc1 orcl.cfg
```

```

PWX-33314 TIMEOUTS configuration parameter is deprecated
PWX-33269 CCL configuration parameter <CHKPT_BASENAME> is deprecated.
PWX-33269 CCL configuration parameter <CHKPT_NUM> is deprecated.
PWX-25404 Processing console program. pwxucdct cmd=report_expired_files cs=C:\Informatica\PowerExchangev.r.m\Resources
\pwxcccl_orcl.cfg
REPORT FOR COMMAND Report Expired Files
-----
CCL FILES
-----
Incarnation: 20130221202420000000 Group #: 1 File Seq#: 1302212029001
Name: C:\Informatica\PowerExchangev.r.m\Resources\ccl\group1\grp1.CND.CP130221.T2029001
Open TimeStamp: 20130221202930000000
Close TimeStamp: 20130221202948000000
Record Count: 840
Commit Count: 60
Has Uncommitted data: No
Status: Closed
Fmt Version: 951
Interest List flags: 1

Restart Information: Sequence Restart
Timestamp
Begin: D4000000CF3E79000000000000000000CF3E77000000000000002F500006E73001000020000 000000CF3E774ED056E4
20130221202929000000
End: D4000000CF3F4A000000000000000000FFFFFFFFFFFFFFFFFFFF000002F5000071850084FFFF0000 000000CF3F494ED056E4
20130221202936000000
```

```

Incarnation: 20130221202420000000 Group #: 1 File Seq#: 1302212029002
Name: C:\Informatica\PowerExchangev.r.m\resources\ccl\group1\grp1.CND.CP130221.T2029002
Open TimeStamp: 20130221202952000000
Close TimeStamp: 20130221202958000000
Record Count: 280
Commit Count: 20
Has Uncommitted data: No
Status: Closed
Fmt Version: 951
Interest List flags: 1

Restart Information: Sequence Restart
Timestamp
Begin: D4000000CF3F6800000000000000000000CF3F67000000000000002F5000071BB001000020000 000000CF3F674ED056E4

```

Restart Information:	Sequence	Restart
Timestamp		
Begin:	D4000000CF3FBB000000000000000000CF3FB9000000000000002F5000072D0014400020000	000000CF3FB94ED056E4
20130221203004000000		
End:	D4000000CF3FFB000000000000FFFFFFFFFFFFFFFF000002F5000073CB017CFFFF0000	000000CF3FFA4ED056E4
20130221203004000000		

The report contains separate sections for the PowerExchange Logger (CCL) configuration incarnation, the groups that are defined in the group definition file that is specified in the `pwxccl.cfg` configuration file, the capture registrations for the tables, and the associated Logger log files. In the CCL Files section, the information that is reported for each log file includes whether the log file is encrypted and the restart tokens.

CHAPTER 21

PWXUCREG - Capture Registration Suspend Utility

This chapter includes the following topics:

- [PWXUCREG Utility Overview, 262](#)
- [PWXUCREG Usage Considerations, 263](#)
- [Supported Operating Systems for the PWXUCREG Utility, 264](#)
- [Suspending Change Capture for Registered Sources Temporarily, 264](#)
- [General Syntax for PWXUCREG Commands, 265](#)
- [Summary of PWXUCREG Commands, 266](#)
- [Global SET_CONTROL_VALUE Parameters, 271](#)
- [Registration-specific Command Parameters, 274](#)
- [Running the PWXUCREG Utility, 276](#)
- [Examples of PWXUCREG Utility Commands, 276](#)

PWXUCREG Utility Overview

Use the PWXUCREG utility to temporarily suspend change capture processing for registered sources. Later, you can use the utility to reactivate the suspended registrations to resume change capture.

This utility has many potential uses. For example, use it to suspend change capture while you make DDL changes such as cascade delete or delete all row changes or corrections to a source or target object.

This utility runs on z/OS only. It processes registrations only for data sources from which the following ECCRs capture data:

- Adabas ECCR
- Datacom table-based ECCR
- IDMS log-based ECCR
- IMS log-based ECCR

You can use the utility to perform the following tasks:

- Suspend capture registrations to temporarily stop change capture activity for registered sources during the suspension window.
- Reactivate suspended capture registrations after a suspension to resume change data capture.

- Display the status setting for capture registrations to verify a status change.
- Skip all change records in the change stream that have timestamps earlier than the current system time when starting change capture for a registration that has been activated for the first time from the PowerExchange Navigator.

When you use the utility to change the registration status, it sets suspension and activation timestamps. The period bounded by the suspension and activation timestamps is called the *suspension window*.

You must refresh the ECCR after a registration status change to have it detect the status change and get the suspension and activation timestamps. During the suspension window, the ECCR discards change records that have a timestamp later than the suspension timestamp. The utility writes message output for registration status changes for audit and monitoring purposes.

The PowerExchange Navigator displays the current registration status in the **Status** field of the Resource Inspector. After the utility processes a PWXUCREG registration suspension request, the Resource Inspector displays the suspension timestamp in current system time in the **Suspend Time** field and displays the value **Suspended** in the **Status** field. After the utility processes a PWXUCREG registration reactivation request, the Resource Inspector displays the activation timestamp in the **Active Time** field and resets the **Status** to **Active**.

PWXUCREG Usage Considerations

Before you begin using the PWXUCREG utility to change the status of registrations, review this list of usage considerations.

- Only one suspension window can be open for a capture registration at a time.
- If you issue a SUSPEND_REGISTRATION command followed by an ACTIVATE_REGISTRATION command and then issue another SUSPEND_REGISTRATION command before the ECCR starts processing the change records later than the first suspension window, unpredictable results might occur. Wait until the ECCR has processed all of the change records with timestamps earlier than the activation timestamp before issuing another suspension request.
- To generate suspension and activation timestamps, the utility uses the current system time at the time when the SUSPEND_REGISTRATION or ACTIVATE_REGISTRATION command is processed, without any adjustment for the local time. These timestamps are included in messages and in the **Suspend Time** and **Active Time** fields of the PowerExchange Navigator Resource Inspector. These timestamps define the start and end of the *suspension window*. The utility uses current system time for the timestamps because the supported database types store timestamps in CDC records in current system time.
- You must issue an ECCR REFRESH command after issuing any PWXUCREG command that changes the registration status. This refresh process enables the ECCR to read the registration information from the CCT data set again to get the suspension and activation timestamps and the new registration status.
- The first time you activate a capture registration in the PowerExchange Navigator, the activation timestamp is not set. The **Active Time** field is blank until you use the PWXUCREG utility to submit a SUSPEND_REGISTRATION command followed by an ACTIVATE_REGISTRATION command.
- If the ECCR ends abnormally and then warm starts within the same suspension window, the utility issues a message when it encounters the first change record in the suspension window to be discarded.
- If you have multiple registrations with the same registration tag name, you must suspend or reactivate each one. The utility cannot process all registrations with the same tag name in a single SUSPEND_REGISTRATION or ACTIVATE_REGISTRATION command.

- When discarding change records for a suspended registrations, the ECCR verifies that the associated UOW started within the suspension window. If the UOW started before the beginning of the suspension window, the ECCR either issues a warning and continues or issues an error message and ends, depending on the ON_SUSPENSION_ERROR_CONTINUE parameter setting in the ECCR configuration file.
- When capturing change records for an activated registration, the ECCR verifies that the associated UOW started after the suspension window closed. If the UOW started before the end of the suspension window, the ECCR either issues a warning and continues or issues an error message and ends, depending on the ON_SUSPENSION_ERROR_CONTINUE parameter setting in the ECCR configuration file.
- To have the ECCR discard change records that have timestamps earlier than the current system time, use the DROP_OLD_REGISTRATION_DATA command. You can issue this command for an Active registration only. This command sets a special suspension window that extends from the earliest point in the logs to the current system time.
- You can cancel a suspension or activation operation before you refresh the ECCR for the registration status change. You might want to cancel a suspension or reactivation request because it was issued at the wrong time related to the database processing or the command input contained errors. The cancel command resets the suspension or activation timestamp.
- In the PowerExchange Navigator, you can change the registration **Status** value of **Suspended** only to **History**. Make this change only if you no longer want to use the registration for change capture. You cannot change a registration **Status** value of **Active** to **Suspended** from the PowerExchange Navigator. You must use the PWXUCREG utility to make this status change.

Supported Operating Systems for the PWXUCREG Utility

The PWXUCREG utility runs on z/OS systems only.

Suspending Change Capture for Registered Sources Temporarily

Use this task flow to suspend change capture processing for registered sources temporarily.

You perform some tasks with the PWXUCREG utility and other tasks outside of the utility on the z/OS system.

Before you begin, ensure that the REFRESH_ALLOWED=Y parameter is specified in the ECCR configuration file. Also, you must have the authority to issue a REFRESH command after each registration status change.

1. Stop database activity for the registered source or sources for which you want to suspend capture registrations.
2. To suspend the capture registrations, use the PWXUCREG utility to issue the SUSPEND_REGISTRATION command.

The suspension window opens. The utility sets the suspension timestamp to the current system time without any adjustment for the local time. Also, the utility issues message PWX-03716 to the DTLLOG log to report the registration status change.

For each suspended registration, the PowerExchange Navigator Resource Inspector displays **Suspended** in the **Status** field and the suspension timestamp in the **Suspend Time** field. The **Suspend Time** value is not adjusted for the local time.

3. For Adabas sources only, perform a PLOG switch.

This step ensures that all of the changes up to the point of the PLOG switch are captured for the active registration.

4. Enter the ECCR REFRESH command with the MVS MODIFY (F) command:

```
F eccr_task_name,REFRESH
```

The ECCR becomes aware of the registration status change and suspension timestamp. When the ECCR encounters the first change record to discard, it issues message PWX-07752. The ECCR discards change records that have a timestamp later than the suspension timestamp.

5. Run the jobs or processes that generate the changes that you do not want to capture for the source or sources that are associated with the suspended registrations.
6. To reactivate the capture registrations, use the PWXUCREG utility to issue the ACTIVATE_REGISTRATION command.

The suspension window closes. The utility sets the activation timestamp to the current system time without any adjustment for the local time. Also, the utility issues message PWX-03716 to the DTLLOG log to report the registration status change.

For each reactivated registration, the PowerExchange Navigator Resource Inspector displays **Active** in the **Status** field and the activation timestamp in the **Active Time** field. The **Active Time** value is not adjusted for the local time.

7. For Adabas sources only, perform a PLOG switch.

This step ensures that all of the changes that occur during the suspension window up to the PLOG switch are discarded for the suspended registration.

8. Enter the ECCR REFRESH command with the MVS MODIFY (F) command again.

The ECCR becomes aware of the registration status change and activation timestamp.

9. Enable database activity to resume on the registered source or sources.

The ECCR starts capturing change records that have timestamps later than the activation timestamp. The ECCR issues message PWX-07753 when it encounters the first change record in the change stream after the end of the suspension window.

Note: You can automate this processing if appropriate for your environment.

General Syntax for PWXUCREG Commands

Use the following general syntax to enter global commands and registration-specific commands in the PWXSYSIN input for a PWXUCREG job:

```
//PWXSYSIN DD *
SET CONTROL_VALUE,global_parameter1=value1,
    global_parameter2=value,
    global_parameter3
;
Registration-specific command primary keyword,
    parameter1=value,
    parameter2=value,
    parameter3=value
;
```

```

    <additional registration-specific commands>
    ;
SET_CONTROL_VALUE,global_parameter1=value2;
Registration-specific command primary keyword,
    parameter1=value,
    parameter4=value
    ;
    <additional registration commands>
    ;
/*

```

The following syntax rules apply:

- A command consists of a primary keyword followed by one or more valid parameters.
- A command ends with a semicolon (;).
- Use a comma (,) to separate a primary keyword from a parameter. Also, use a comma between parameters. Do not use a comma after the last parameter in a command, before the ending semicolon.
- In a SET_CONTROL_VALUE command, you can specify one or more global parameters. Global parameters are optional, but you must include at least one global parameter in the command. Alternatively, you can specify a separate SET_CONTROL_VALUE statement for each parameter.
- In a registration-specific command, you can enter multiple parameters. Some parameters are required to identify the registrations to process, and other parameters are optional.
- The utility parses and executes commands in the input stream one at a time from top to bottom.
- You can repeat the SET_CONTROL_VALUE command in the input stream with different parameters or with the same parameters but different values for a subsequent set of registration-specific commands.
- You can include parameters in a registration-specific command that overwrite corresponding global parameters in the preceding SET_CONTROL_VALUE statement. For example, include the REGISTRATION_LOCATION parameter in a registration-specific command to overwrite the GLOBAL_REGISTRATION_LOCATION parameter in the previous SET_CONTROL_VALUE command.

Summary of PWXUCREG Commands

Use the summary of PWXUCREG utility commands to determine which command keyword and parameters you want to use.

The SET_CONTROL_VALUE keyword sets global parameter values such as the user ID and password for subsequent commands in the PWXSYSIN input stream for the PWXUCREG job. The registration-specific commands apply to the registration or registrations that match the selection criteria that you enter in parameters such as those for the database instance and registration name.

The following table describes the SET_CONTROL_VALUE global command and each registration-specific command, including the primary keyword and associated parameters:

Command Keyword	Description	Parameters ¹
SET_CONTROL_VALUE	<p>Sets global parameter values that apply to subsequent registration-specific commands in the PWXSYSIN input of the PWXUCREG JCL. Use this command if you run PWXUCREG jobs that contain multiple commands and want to define common parameter values once for a set of commands.</p> <p>You can override any global value in a subsequent registration-specific command or by specifying another SET_CONTROL_VALUE command later in the input stream.</p> <p>You must specify at least one parameter for this command.</p>	<ul style="list-style-type: none"> - [DISPLAY_REGISTRATION_AFTER_COMMAND] - [DISPLAY_REGISTRATION_BEFORE_COMMAND] - [GENERIC] - [GLOBAL_REGISTRATION_LOCATION] - [GLOBAL_USER] - [GLOBAL_EPWD] - [GLOBAL_PWD] - [SYSTEM_CONSOLE_MESSAGES_COMMAND] - [SYSTEM_CONSOLE_MESSAGES_DISPLAY] - [SHOW_EXPANDED_STATEMENT]
ACTIVATE_REGISTRATION	<p>Reactivates a capture registration that has a status of Suspended so that the ECCR resumes change capture for the registered source. Also sets an activation timestamp in current system time, not adjusted for local time, to indicate the end of the suspension window.</p> <p>You can issue this command only for registrations that were previously suspended with the SUSPEND_REGISTRATION command.</p> <p>This command can apply to multiple registrations if you specify GENERIC=Y and enter the asterisk (*) wildcard, or a string followed by the wildcard, in the REGISTRATION_NAME or DATABASE_INSTANCE parameter.</p>	<ul style="list-style-type: none"> - REGISTRATION_NAME - DATABASE_INSTANCE - [DATABASE_TYPE] - Recommended - [EPWD] - [GENERIC] - [PWD] - [REGISTRATION_LOCATION] - [USER] - [VALIDATE]

Command Keyword	Description	Parameters ¹
CANCEL_ACTIVATE_REGISTRATION	<p>Cancels a previous ACTIVATE_REGISTRATION request before you refresh the ECCR for the activation. Also sets the registration status back to Suspended.</p> <p>This command can apply to multiple registrations if you specify GENERIC=Y and enter the asterisk (*) wildcard, or a string followed by the wildcard, in the REGISTRATION_NAME or DATABASE_INSTANCE parameter.</p> <p>Tip: To cancel the reactivation action for all registrations that were specified in the previous ACTIVATE_REGISTRATION command, specify the same parameter values as in the ACTIVATE_REGISTRATION command.</p>	<ul style="list-style-type: none"> - REGISTRATION_NAME - DATABASE_INSTANCE - [DATABASE_TYPE] - Recommended - [EPWD] - [GENERIC] - [PWD] - [REGISTRATION_LOCATION] - [USER] - [VALIDATE]
CANCEL_SUSPEND_REGISTRATION	<p>Cancels a previous SUSPEND_REGISTRATION command before you refresh the ECCR. Also resets the registration status to Active and resets the start and end times for the suspension window.</p> <p>This command can apply to multiple registrations if you specify GENERIC=Y and enter the asterisk (*) wildcard, or a string followed by the wildcard, in the REGISTRATION_NAME or DATABASE_INSTANCE parameter.</p> <p>Tip: To cancel the suspension action for all registrations that were specified in the previous SUSPEND_REGISTRATION command, specify the same parameter values as in the SUSPEND_REGISTRATION command.</p>	<ul style="list-style-type: none"> - REGISTRATION_NAME - DATABASE_INSTANCE - [DATABASE_TYPE] - Recommended - [EPWD] - [GENERIC] - [PWD] - [REGISTRATION_LOCATION] - [USER] - [VALIDATE]

Command Keyword	Description	Parameters ¹
DISPLAY_REGISTRATION	<p>Displays registration status information before or after another command that changes registration status so that you can verify the status change. This information includes the current registration status setting and the activation and suspension timestamps. The timestamps are in current system time and not adjusted for local time.</p> <p>This command does not support the GENERIC parameter or the global GENERIC parameter of the SET_CONTROL_VALUE command. This command can display status information for multiple registrations without the GENERIC parameter.</p> <p>Tip: Instead of specifying this command in the JCL input stream multiple times, you can specify the global SET_CONTROL_VALUE keyword with the DISPLAY_REGISTRATION_BEFORE_COMMAND and DISPLAY_REGISTRATION_AFTER_COMMAND parameters. These global parameters automatically display the registration status information before and after each command that changes the registration status.</p>	<ul style="list-style-type: none"> - REGISTRATION_NAME - DATABASE_INSTANCE - [DATABASE_TYPE] - Recommended - [EPWD] - [PWD] - [REGISTRATION_LOCATION] - [USER] - [VALIDATE]

Command Keyword	Description	Parameters ¹
DROP_OLD_REGISTRATION_DATA	<p>Sets a special suspension window that enables you to begin change capture for a registration from the current system time. The suspension window extends from the earliest available point in the change stream to the current system time. The ECCR discards change records that have timestamps within the suspension window. Often, this command is used for new registrations that were just activated for the first time from the PowerExchange Navigator.</p> <p>This command can apply to multiple registrations if you specify GENERIC=Y and enter the asterisk (*) wildcard, or a string followed by the wildcard, in the REGISTRATION_NAME or DATABASE_INSTANCE parameter.</p>	<ul style="list-style-type: none"> - REGISTRATION_NAME - DATABASE_INSTANCE - [DATABASE_TYPE] - Recommended - [EPWD] - [GENERIC] - [PWD] - [REGISTRATION_LOCATION] - [USER] - [VALIDATE]
RESET_SUSPENSION_WINDOW	<p>Clears any activation and suspension timestamps that define the current suspension window and resets the registration status to Active.</p> <p>This command can apply to multiple registrations if you specify GENERIC=Y and enter the asterisk (*) wildcard, or a string followed by the wildcard, in the REGISTRATION_NAME or DATABASE_INSTANCE parameter.</p>	<ul style="list-style-type: none"> - REGISTRATION_NAME - DATABASE_INSTANCE - [DATABASE_TYPE] - Recommended - [EPWD] - [GENERIC] - [PWD] - [REGISTRATION_LOCATION] - [USER] - [VALIDATE]

Command Keyword	Description	Parameters ¹
SUSPEND_REGISTRATION	<p>Suspends a capture registration that has a status of Active so that the ECCR stops capturing changes for the registered source. Also sets a suspension timestamp in current system time, not adjusted for local time, to indicate the start of the suspension window.</p> <p>This command can apply to multiple registrations if you specify GENERIC=Y and enter the asterisk (*) wildcard, or a string followed by the wildcard, in the REGISTRATION_NAME or DATABASE_INSTANCE parameter.</p>	<ul style="list-style-type: none"> - REGISTRATION_NAME - DATABASE_INSTANCE - [DATABASE_TYPE] - Recommended - [EPWD] - [GENERIC] - [PWD] - [REGISTRATION_LOCATION] - [USER] - [VALIDATE]
1. Parameters enclosed in square brackets [] are optional.		

Global SET_CONTROL_VALUE Parameters

You can specify one or more global parameters in a SET_CONTROL_VALUE command that you include in the PWXSYSIN input for a PWXUCREG job.

The global parameters apply to the subsequent registration-specific commands in the PWXSYSIN input, unless you override the global parameter value in one of the following ways:

- You specify a subsequent registration-specific command with the corresponding parameter, if available.
- You specify another SET_CONTROL_VALUE command with a different global parameter value later in the input stream.

All of the global parameters are optional but you must specify at least one in a SET_CONTROL_VALUE command.

In the following parameter descriptions, curly brackets indicate that one of the options must be entered, and underlining indicates the default value.

Parameter Descriptions

DISPLAY_REGISTRATION_AFTER_COMMAND={N|Y}

Displays information about the capture registration or registrations that were processed by the preceding registration-specific command that changed the registration status. This information includes the registration current status setting and activation and suspension timestamps. You can use this information to verify that the preceding command correctly changed the status. Options are:

- **N**. Do not display registration information after each command that changes the registration status.
- **Y**. Display registration information after each command that changes the registration status.

Default is Y. If you accept the default value, you do not need to specify the DISPLAY_REGISTRATION command in the PWXUCREG JCL to display registration status information after a command, unless you want to override this global parameter setting.

You can use this parameter with the DISPLAY_REGISTRATION_BEFORE_COMMAND parameter to display registration status information before and after a command that changes the status.

DISPLAY_REGISTRATION_BEFORE_COMMAND={N|Y}

Displays information about the capture registration or registrations that will be processed by a subsequent registration-specific command that changes the registration status. This information includes the registration current status setting and activation and suspension timestamps. You can use this information to verify that the command correctly changes the status for the original value to the target value. Options are:

- **N.** Do not display registration information before each command that changes the registration status.
- **Y.** Display registration information before each command that changes the registration status.

Default is N. If you accept the default value, you can still specify the DISPLAY_REGISTRATION command in the PWXUCREG JCL to display registration status information before a particular command.

You can use this parameter with the DISPLAY_REGISTRATION_AFTER_COMMAND parameter to display registration status information before and after a command that changes the status.

GENERIC={N|Y}

Enables you to issue registration-specific commands that change the registration status for multiple registrations. You must also specify masks that contain the asterisk (*) wildcard in certain parameters of the registration-specific commands, such as REGISTRATION_NAME and DATABASE_INSTANCE.

This parameter is not required or supported for the DISPLAY_REGISTRATION command, which is generic by default.

You can override this global setting for a particular registration-specific command by including the GENERIC parameter in the command.

GLOBAL_EPWD=*encrypted_password*

Specifies an encrypted password to use with the user ID that is specified in the associated GLOBAL_USER parameter.

Consult with your system security administrator to determine the requirements for a user ID and a password or encrypted password based on your system security and the SECURITY statement in the DBMOVER configuration member.

Tip: You can use the PowerExchange Navigator to create an encrypted password. Click **File > Encrypt**.

If you specify this parameter, do not also specify the GLOBAL_PWD parameter. If you specify both, the GLOBAL_EPWD parameter takes precedence. Use the GLOBAL_EPWD parameter if you are not allowed to store passwords in readable format.

You can override this value for a particular registration-specific command by including the EPWD parameter in the command.

GLOBAL_PWD=*password*

Specifies a nonencrypted password to use with the user ID that is specified in the associated GLOBAL_USER parameter. This value is case-sensitive.

Consult with your system security administrator to determine the requirements for a user ID and a password or encrypted password based on your system security and the SECURITY statement in the DBMOVER configuration member.

If you specify this parameter, do not also specify the GLOBAL_EPWD parameter. If you specify both, the GLOBAL_EPWD parameter takes precedence.

You can override this value for a particular registration-specific command by including the PWD parameter in the command.

GLOBAL_REGISTRATION_LOCATION={hlq.data_set_name|local}

Specifies the location of the VSAM CCT data set that contains the capture registrations. Default value is "local."

Usually, the default value of "local" is acceptable because the capture registrations must reside on the z/OS source system where the PWXUCREG utility runs.

You can override this value for a particular registration-specific command by including the REGISTRATION_LOCATION parameter in the command.

GLOBAL_USER=user_id

Specifies a user ID that has the authority to access capture registrations in the CCT data set on the source system. This value is case-sensitive.

Consult with your system security administrator to determine the requirements for a user ID and a password or encrypted password based on your system security and the SECURITY statement in the DBMOVER configuration member.

If you specify this parameter, also specify either the GLOBAL_EPWD or GLOBAL_PWD parameter.

You can override the global user ID for a particular registration-specific command by including the USER parameter in the command.

SHOW_EXPANDED_STATEMENT={N|Y}

Displays the input command statements in an expanded format that includes the parameters that you specified in the PWXUCREG job and the other valid parameters that you did not specify but are in effect with their default values or an asterisk (*) wildcard entry.

SYSTEM_CONSOLE_MESSAGES_COMMAND={N|Y}

Controls whether the utility routes message output from registration-specific commands that change the registration status to the z/OS system console as well as to the DTLLOG log. If you enter Y, messages are sent to the system console and the DTLLOG log. If you accept the default value of N, the messages are sent only to the DTLLOG log.

SYSTEM_CONSOLE_MESSAGES_DISPLAY={N|Y}

Controls whether the utility routes message output from an explicit or automatic display registration request to the z/OS system console as well as to the DTLLOG log. These requests result from a DISPLAY_REGISTRATION command or a SET_CONTROL_VALUE command that includes the global DISPLAY_REGISTRATION_AFTER_COMMAND or DISPLAY_REGISTRATION_BEFORE_COMMAND parameter. If you enter Y, the messages are sent to the system console and the DTLLOG log. If you accept the default value of N, the messages are sent only to the DTLLOG log.

Registration-specific Command Parameters

You can specify multiple parameters for any registration-specific command that you include in the PWXSYSIN input for a PWXUCREG job.

The DATABASE_INSTANCE and REGISTRATION_NAME parameters are required.

Registration-specific command parameters override any corresponding global parameters.

In the following parameter descriptions, curly brackets indicate that one of the options must be entered, and underlining indicates the default value.

Parameter Descriptions

DATABASE_TYPE=type

Recommended. Specifies the type of source database and CDC that is associated with the registration or registrations that the PWXUCREG command processes. Options are:

- **ADA**. For Adabas.
- **DCM**. For Datacom table-based.
- **IDL**. For IDMS log-based.
- **IMS**. For IMS log-based. If you try to process registrations for IMS synchronous sources, the PWXUCREG utility rejects the registrations and ends with error message PWX-03723.

You can use the asterisk (*) wildcard, or a string followed by the wildcard such as B08*, in this parameter.

DATABASE_INSTANCE=instance_id

Required. Specifies one of the following values that is defined for the registration group in the PowerExchange Navigator:

- For Adabas, the collection identifier.
- For Datacom, the MUF name.
- For IDMS, the LOGSID identifier.
- For IMS, the RECON identifier.

This value is case-sensitive. You can use the asterisk (*) wildcard, or a string followed by the wildcard, in this parameter.

EPWD=encrypted_password

Specifies an encrypted password to use with the user ID that is specified in the associated USER parameter.

Consult with your system security administrator to determine the requirements for a user ID and a password or encrypted password based on your system security and the SECURITY statement in the DBMOVER configuration member.

Tip: You can use the PowerExchange Navigator to create an encrypted password. Click **File > Encrypt**.

If you specify this parameter, do not also specify the PWD parameter. If you specify both, the EPWD parameter takes precedence. Use the EPWD parameter if you are not allowed to store passwords in readable format.

Overrides the GLOBAL_EPWD parameter in a preceding SET_CONTROL_VALUE command, if specified.

GENERIC={N|Y}

Optional. Enables you to issue registration-specific commands that change the registration status for multiple registrations that match a wildcard mask if you specify Y. You must also specify the asterisk (*) wildcard, or a string followed by the wildcard such as b80*, in one or more of the following registration-specific parameters: REGISTRATION_NAME, DATABASE_INSTANCE, and DATABASE_TYPE.

Default is N.

This parameter is not supported for any explicit DISPLAY_REGISTRATION command or automatic display operation based on the global DISPLAY_REGISTRATION_BEFORE_COMMAND and DISPLAY_REGISTRATION_AFTER_COMMAND parameters.

Overrides the global GENERIC parameter in a preceding SET_CONTROL_VALUE command.

PWD=password

Specifies a nonencrypted password to use with the user ID that is specified in the associated GLOBAL_USER parameter. This value is case-sensitive.

Consult with your system security administrator to determine the requirements for a user ID and a password or encrypted password based on your system security and the SECURITY statement in the DBMOVER configuration member.

If you specify this parameter, do not also specify the EPWD parameter. If you specify both, the EPWD parameter takes precedence.

Overrides the GLOBAL_PWD parameter in a preceding SET_CONTROL_VALUE command, if specified.

REGISTRATION_LOCATION={hlq.data_set_name|local}

Optional. Specifies the location of the VSAM CCT data set that contains the capture registrations. Default value is "local."

Usually, the default value of "local" is acceptable because the capture registrations must reside on the z/OS source system where the PWXUCREG utility runs.

Overrides the GLOBAL_REGISTRATION_LOCATION parameter in a preceding SET_CONTROL_VALUE command, if specified.

REGISTRATION_NAME=user_registration_name

Required. Specifies the user-defined name for the registration that is defined for the capture registration in the PowerExchange Navigator. This value is case-sensitive.

You can use the asterisk (*) wildcard, or a string followed by the wildcard, in this parameter.

USER=user_id

Optional. Specifies a user ID that has the authority to access capture registrations in the CCT data set on the source system.

Consult with your system security administrator to determine the requirements for a user ID and a password or encrypted password based on your system security and the SECURITY statement in the DBMOVER configuration member.

This user ID value is case-sensitive.

If you specify this parameter, also specify either the EPWD or PWD parameter. If you specify both, the EPWD parameter takes precedence.

Overrides the GLOBAL_USER parameter in a preceding SET_CONTROL_VALUE command, if specified.

VALIDATE={N|Y}

Optional. Validates the command syntax and reads the registrations that match the selection criteria to ensure that the command can process them, if you specify Y. This parameter does not actually run the command to change the registration status. Default is N, which disables validation.

Running the PWXUCREG Utility

After you define the JCL for the PWXUCREG job, you can manually submit the job or schedule it to run in an automated manner as part of a batch job.

The following JCL includes the basic statements:

```
//PWXURACT JOB 'PWXUCREG',MSGLEVEL=(1,1),MSGCLASS=X,NOTIFY=&SYSUID,
//          CLASS=A
//LIBSRCH JCLLIB ORDER=DTLUSR.V951.RUNLIB
//          SET HLQ=DTLUSR.V951
//          SET RUNLIB=DTLUSR.V951.RUNLIB
//STEP1    EXEC PGM=PWXUCREG
//STEPLIB DD DISP=SHR,DSN=&HLQ..LOADLIB
//*
//DTLAMCPR DD DISP=SHR,DSN=&HLQ..CCT
//DTLMSG DD DISP=SHR,DSN=&HLQ..DTLMSG
//DTLCFG DD DISP=SHR,DSN=&RUNLIB(DBMOVER)
//DTLKEY DD DISP=SHR,DSN=&RUNLIB(LICENSE)
//DTLSGN DD DISP=SHR,DSN=&RUNLIB(SIGNON)
//DTLLOG DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//PWXSYSIN DD *
          <pwxucreg commands and parameters>
/*
```

Examples of PWXUCREG Utility Commands

Use the example PWXUCREG commands to learn how to code simple commands and recognize their message output in the job log.

Example 1. Suspending a Capture Registration

You need to perform a cascade delete operation for a single registered source but do not want the ECCR to capture these deletes.

To stop change capture temporarily, you plan to suspend the capture registration named b800tbl that is associated with the source. You run a PWXUCREG job that contains a SUSPEND_REGISTRATION command. Later, you will reactivate the capture registration to resume capture processing.

Use the following JCL to run a PWXUCREG job that suspends the capture registration:

```
//PWXURSUS JOB 'PWXUCREG',MSGLEVEL=(1,1),MSGCLASS=X,NOTIFY=&SYSUID,
//          CLASS=A
//LIBSRCH JCLLIB ORDER=DTLUSR.V951.RUNLIB
//          SET HLQ=DTLUSR.V951
//          SET RUNLIB=DTLUSR.V951.RUNLIB
//STEP1    EXEC PGM=PWXUCREG
```

```
//STEPLIB DD DISP=SHR,DSN=&HLQ..LOADLIB
//*
//DTLAMCPR DD DISP=SHR,DSN=&HLQ..CCT
//DTLMSG DD DISP=SHR,DSN=&HLQ..DTLMSG
//DTLCFG DD DISP=SHR,DSN=&RUNLIB (DBMOVER)
//DTLKEY DD DISP=SHR,DSN=&RUNLIB (LICENSE)
//DTLSGN DD DISP=SHR,DSN=&RUNLIB (SIGNON)
//DTLLOG DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//PWXSYSIN DD *
SET CONTROL VALUE,SYSTEM_CONSOLE_MESSAGES_COMMAND=Y;
SUSPEND_REGISTRATION,
DATABASE_INSTANCE=R11G4,
REGISTRATION_NAME=b800tb1
/*
```

The JCL includes the global SYSTEM_CONSOLE_MESSAGES_COMMAND=Y parameter in the SET_CONTROL_VALUE command to write messages related to suspension processing to the z/OS system console.

If the JCL is successfully processed, the following messages are printed to both the job log and z/OS system console:

```
16.22.35 JOB03118 ---- TUESDAY, 23 OCT 2012 ----
16.22.35 JOB03118 IRR010I USERID DTLUSR IS ASSIGNED TO THIS JOB.
16.22.35 JOB03118 ICH70001I DTLUSR LAST ACCESS AT 14:01:58 ON TUESDAY, OCTOBER 23, 2012
16.22.35 JOB03118 EHASP373 PWXURSUS STARTED - INIT 3 - CLASS A - SYS MHZ1
16.22.35 JOB03118 IEF403I PWXURSUS - STARTED - TIME=16.22.35
16.22.37 JOB03118 PWX-03716 PWXUCREG: Registration "b800tb1", version "1", instance "R11G4", status changed from "Active" to
                  "Suspended" .
16.22.37 JOB03118 - --TIMINGS (MINS.)-- -----PAGING COUNTS-----
16.22.37 JOB03118 -STEPNAME PROCSTEP RC EXCP CONN TCB SRB CLOCK SERV WORKLOAD PAGE SWAP VIO SWAPS
16.22.37 JOB03118 -STEP1 00 1020 417 .00 .00 .0 5542 BATCH 0 0 0 0
16.22.38 JOB03118 IEF404I PWXURSUS - ENDED - TIME=16.22.38
16.22.38 JOB03118 -PWXURSUS ENDED. NAME- TOTAL TCB CPU TIME= .00 TOTAL ELAPSED TIME= .0
16.22.38 JOB03118 EHASP395 PWXURSUS ENDED
----- JES2 JOB STATISTICS -----
 23 OCT 2012 JOB EXECUTION DATE
 24 CARDS READ
142 SYSOUT PRINT RECORDS
 0 SYSOUT PUNCH RECORDS
 8 SYSOUT SPOOL KBYTES
 0.04 MINUTES EXECUTION TIME

PWX-33314 TIMEOUTS configuration parameter is
deprecated

PWX-15799 DD:PWXSYSIN <> PARM INPUT FILE:
START>>> .
PWX-15799
SET CONTROL_VALUE,SYSTEM_CONSOLE_MESSAGES_COMMAND=Y;.
PWX-15799
SUSPEND_REGISTRATION,.
PWX-15799
DATABASE_INSTANCE=R11G4,.
PWX-15799
REGISTRATION_NAME=b800tb1.
PWX-15799 ;.

PWX-15799 DD:PWXSYSIN <> PARM INPUT FILE:
END(COMPLETE).

PWX-03716 PWXUCREG: Registration "b800tb1", version "1", instance "R11G4", status changed from "Active" to
"Suspended".

PWX-03717 PWXUCREG: Number of registrations processed
1 .

PWX-03712 PWXUCREG: Registration "b800tb1", type "DCM", instance "R11G4", version "1", current status
"S".
PWX-03713 PWXUCREG: Registration "b800tb1", suspended at "2012/10/23 16:22:37.235636", current time "2012/10/23
16:22:37.727037".
```

```
PWX-03714 PWXUCREG: Registration "b800tbl", activated at "2012/10/19 09:46:26.007478", current time "2012/10/23 16:22:37.727037".
```

```
PWX-03724 PWXUCREG: Number of registrations displayed 1.
```

Message PWX-03716 indicates that the utility successfully changed the registration status to Suspended. Messages PWX-03712 through PWX-03714 are printed because you accepted the default value of Y for global DISPLAY_REGISTRATION_AFTER_COMMAND parameter. These messages report registration information such as the current registration status, the suspension timestamp in current system time, and the activation timestamp from a previous ACTIVATE_REGISTRATION command.

Example 2. Reactivating a Capture Registration

You previously suspended a capture registration for a source to ignore cascade delete operations and now want to reactivate the registration to begin capturing changes again.

To close the suspension window and resume change capture for the capture registration named b800tbl, you run a PWXUCREG job that contains an ACTIVATE_REGISTRATION command.

Use the following JCL to run a PWXUCREG job that suspends the capture registration:

```
//PWXURACT JOB 'PWXUCREG',MSGLEVEL=(1,1),MSGCLASS=X,NOTIFY=&SYSUID,
//          CLASS=A
//LIBSRCH JCLLIB ORDER=DTLUSR.V951.RUNLIB
//          SET HLQ=DTLUSR.V951
//          SET RUNLIB=DTLUSR.V951.RUNLIB
//STEP1 EXEC PGM=PWXUCREG
//STEPLIB DD DISP=SHR,DSN=&HLQ..LOADLIB
//*
//DTLAMCPR DD DISP=SHR,DSN=&HLQ..CCT
//DTLMSG DD DISP=SHR,DSN=&HLQ..DTLMSG
//DTLCFG DD DISP=SHR,DSN=&RUNLIB(DBMOVER)
//DTLKEY DD DISP=SHR,DSN=&RUNLIB(LICENSE)
//DTLSGN DD DISP=SHR,DSN=&RUNLIB(SIGNON)
//DTLLOG DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//PWXSYSIN DD *
SET_CONTROL_VALUE,SYSTEM_CONSOLE_MESSAGES_COMMAND=Y;
ACTIVATE_REGISTRATION,
DATABASE_INSTANCE=R11G4,
REGISTRATION_NAME=b800tbl
/*
```

The JCL includes the global SYSTEM_CONSOLE_MESSAGES_COMMAND=Y parameter in the SET_CONTROL_VALUE command to write messages related to reactivation processing to the z/OS system console.

If the JCL is successfully processed, the following messages are printed to both the job log and z/OS system console:

```
16.22.57 JOB03119 ---- TUESDAY, 23 OCT 2012 ----
16.22.57 JOB03119 IRR010I USERID DTLUSR IS ASSIGNED TO THIS JOB.
16.22.58 JOB03119 ICH70001I DTLUSR LAST ACCESS AT 16:22:35 ON TUESDAY, OCTOBER 23, 2012
16.22.58 JOB03119 £HASP373 PWXURACT STARTED - INIT 3 - CLASS A - SYS MHZ1
16.22.58 JOB03119 IEF403I PWXURACT - STARTED - TIME=16.22.58
16.22.59 JOB03119 PWX-03716 PWXUCREG: Registration "b800tbl", version "1", instance "R11G4", status changed from "Suspended" to
"Active" .
16.23.00 JOB03119 - --TIMINGS (MINS.)-- -----PAGING COUNTS-----
16.23.00 JOB03119 -STEPNAME PROCSTEP RC EXCP CONN TCB SRB CLOCK SERV WORKLOAD PAGE SWAP VIO SWAPS
16.23.00 JOB03119 -STEP1 00 1019 346 .00 .00 .0 5350 BATCH 0 0 0 0
16.23.00 JOB03119 IEF404I PWXURACT - ENDED - TIME=16.23.00
16.23.00 JOB03119 -PWXURACT ENDED. NAME- TOTAL TCB CPU TIME= .00 TOTAL ELAPSED TIME= .0
16.23.00 JOB03119 £HASP395 PWXURACT ENDED
----- JES2 JOB STATISTICS -----
23 OCT 2012 JOB EXECUTION DATE
24 CARDS READ
142 SYSOUT PRINT RECORDS
0 SYSOUT PUNCH RECORDS
8 SYSOUT SPOOL KBYTES
```

0.03 MINUTES EXECUTION TIME

PWX-33314 TIMEOUTS configuration parameter is deprecated

```
PWX-15799 DD:PWXSYSIN <> PARM INPUT FILE: START>>> .
PWX-15799 SET_CONTROL_VALUE,SYSTEM_CONSOLE_MESSAGES_COMMAND=Y;.
PWX-15799 ACTIVATE_REGISTRATION,.
PWX-15799 DATABASE_INSTANCE=R11G4,.
PWX-15799 REGISTRATION_NAME=b800tbl.
PWX-15799 ;.
PWX-15799 DD:PWXSYSIN <> PARM INPUT FILE: END(COMPLETE).
```

PWX-03716 PWXUCREG: Registration "b800tbl", version "1", instance "R11G4", status changed from "Suspended" to "Active" .

PWX-03717 PWXUCREG: Number of registrations processed 1 .

```
PWX-03712 PWXUCREG: Registration "b800tbl", type "DCM", instance "R11G4", version "1", current status "A".
PWX-03713 PWXUCREG: Registration "b800tbl", suspended at "2012/10/23 16:22:37.235636", current time "2012/10/23 16:23:00.219928".
PWX-03714 PWXUCREG: Registration "b800tbl", activated at "2012/10/23 16:22:59.789082", current time "2012/10/23 16:23:00.219928".
```

PWX-03724 PWXUCREG: Number of registrations displayed
1.

Message PWX-03716 indicates that the utility successfully changed the registration status to Activated. Messages PWX-03712 through PWX-03714 are printed because you accepted the default value of Y for global DISPLAY_REGISTRATION_AFTER_COMMAND parameter. These messages report registration information such as the current registration status, the activation timestamp, and the suspension timestamp from the last SUSPEND_REGISTRATION command.

CHAPTER 22

PWXUCRGP - Capture Registrations Print Utility

This chapter includes the following topics:

- [PWXUCRGP Utility Overview, 280](#)
- [Supported Operating Systems for the PWXUCRGP Utility, 280](#)
- [Control Statement Syntax for the PWXUCRGP Utility, 281](#)
- [PWXUCRGP Parameters, 281](#)
- [Running the PWXUCRGP Utility, 284](#)
- [PWXUCRGP Report Levels of Detail, 284](#)
- [Examples of PWXUCRGP Reports, 290](#)

PWXUCRGP Utility Overview

The PWXUCRGP utility reports the capture registration information that is stored in the CCT file, for all registrations or a subset of registrations. You can filter the list of registrations based on database type, instance, name, registration name, or registration status. You can also report information for a specific capture registration.

The reports contain information that you can use in various situations such as migrating registrations to another system or diagnosing registration-related problems.

The report output is written to a text file on the local machine, which you specify in the syntax. You can control the report level of detail and format.

Supported Operating Systems for the PWXUCRGP Utility

The PWXUCRGP utility runs on computers that have the following types of operating systems:

- Linux
- UNIX

- Windows

For information about the supported versions of these operating systems, see the *PowerExchange Installation and Upgrade Guide*.

Control Statement Syntax for the PWXUCRGP Utility

Use the following syntax to specify control statements for the PWXUCRGP utility:

```
PWXUCRGP OUTPUT_FILE=file_name
[LOCATION={node_name|local}]
[OVERRIDE_CCT_FILE=file_name]
[UID=user_name]
[EPWD=encrypted_password|PWD=password]
[CRNAME=registration_name]
[DBTYPE=database_type]
[INSTANCE=instance]
[REPORT_LEVEL={SINGLELINE|SUMMARY|COLUMNS}]
[REPORT_SEQUENCE={TAG|TABLE|CRNAME}]
[STATUS=registration_status]
[TRACING={N|Y}]
```

PWXUCRGP Parameters

The utility supports the following parameters:

OUTPUT_FILE=file_name

Required. The name of the output file for the report, such as `my_registrations.txt`. The output file is a standard text file on the local Linux, UNIX, or Windows machine. The target location for the output file cannot be on i5/OS or z/OS.

LOCATION={node_name|local}

The location of the CCT file that contains the registration information. You can specify **local** if the CCT file is on the machine where the utility runs, or you can specify the node name of a remote machine where a PowerExchange Listener runs. If the location is a remote Listener node, and the SECURITY parameter in the DBMOVER configuration file is set to level 1 or 2, a user ID and either a password or an encrypted password are required.

Default is **local**.

OVERRIDE_CCT_FILE=file_name

The name of the CCT file that contains the registration information, if different than the current CCT file for the PowerExchange Listener specified in the LOCATION parameter. For example, if you use the DTLURDMO utility to copy registrations to a specific CCT file name, you can use this parameter to report information from that file.

UID=user_name

A user name that allows access to the remote PowerExchange Listener that is identified by the LOCATION parameter. The requirement for the user name depends the value of the SECURITY statement in the PowerExchange DBMOVER configuration file associated with the remote Listener.

For a source on a supported Linux, UNIX, and Windows system, if you enabled PowerExchange LDAP user authentication and, if applicable, disabled relational pass-through authentication, the user name is the enterprise user name. For more information, see the *PowerExchange Reference Manual*.

EPWD=encrypted_password

An encrypted password for the user who is specified in the UID parameter.

If the utility accesses a remote i5/OS or z/OS location, you can enter an encrypted PowerExchange passphrase instead of an encrypted password. Do not encrypt a passphrase that contains invalid characters, such as double-quotation marks, single quotation marks, or currency symbols.

Do not specify both the EPWD and PWD parameters. Use EPWD if you are not allowed to store passwords in a readable format.

PWD=password

A clear-text password for the user who is specified in the UID parameter. If the utility accesses a remote i5/OS or z/OS location, you can enter a valid PowerExchange passphrase instead of a password.

Do not specify both the PWD and EPWD parameters .

CRNAME=registration_name

The registration name that is used to filter the registrations that are included in the report. The name can be up to eight lowercase characters and can include a trailing asterisk if multiple registration names use the same format. For example, you can specify *int2** to include the registration names *int2a*, *int2b*, and *int2c*. The registration name can be user defined when a capture registration is added in the PowerExchange Navigator or generated by the DTLUCBRG utility.

DBTYPE=database_type

A three-character keyword for the source database type. This value is used to filter registrations for which the utility reports information. Options are:

- Asterisk (*) wildcard only, for all supported types of databases
- **ADA** for Adabas
- **AS4** for DB2 for i (i5/OS)
- **DB2** for DB2 for z/OS
- **DCM** for Datacom.
- **IDM** for log-based IDMS
- **IMS** for IMS
- **MSS** for Microsoft SQL Server
- **MYS** for MySQL
- **ORA** for Oracle
- **PGS** for PostgreSQL
- **UDB** for DB2 for Linux, UNIX, and Windows
- **VSM** for VSAM

Default is the asterisk (*) wildcard.

Note: Use **DB2** only for DB2 for z/OS.

INSTANCE=instance_name

An instance name for the specified database type. If you specify the DBTYPE parameter, you can use the INSTANCE parameter to further filter the registrations for which information is reported to a specific *instance_name*.

REPORT_LEVEL={SINGLELINE|SUMMARY|COLUMNS}

The level of detail to include in the report. Options are:

- **SINGLELINE.** Reports a single line of information for each capture registration that is included in the report. This information is also included in the SUMMARY and COLUMNS report outputs. For more information, see [“Single-Line Report Content” on page 284](#).
- **SUMMARY.** Reports summary information about each capture registration in the report. A SUMMARY report includes the same information as a SINGLELINE report and additionally includes the user name associated with the registration group, the most recent update date, the PowerExchange version of the system that updated the registration, edit sequence, registration type, and data columns. For more information, see [“Summary Report Content” on page 285](#).
- **COLUMNS.** Reports information about each column in the capture registration. A COLUMN report includes the same information as a SUMMARY report and additionally includes detailed information about each column in the capture registration. For more information, see [“Column-Level Report Content” on page 287](#).

Default is SINGLELINE. For more information, see [“PWXUCRGP Report Levels of Detail” on page 284](#).

REPORT_SEQUENCE={TAG|TABLE|CRNAME}

A secondary level of sorting information in the report. Each report is sorted first by database type and instance and then by the criterion in this parameter. Options are:

- **TAG.** Sort by registration tag name.
- **TABLE.** Sort by schema and table name.
- **CRNAME.** Sort by CRNAME.

Default is **TAG**.

STATUS=registration_status

A registration status to use for filtering the capture registrations for which to report information. Options are:

- Asterisk (*) wildcard only. Include capture registrations of all statuses.
- **A.** Include only capture registrations with an Active status.
- **H.** Include only capture registrations with a History status.
- **I.** Include only capture registrations with an Inactive status.
- **S.** Include only capture registrations with a Suspended status.

Default is asterisk (*).

TRACING={Y|N}

Turns on the UCRGP trace filter for the utility without having to edit the TRACING statement in the PowerExchange DBMOVER configuration file. Options are:

- **Y.** Run the trace filter for the utility.
- **N.** Do not run the trace filter for the utility.

Default is **N**. Set TRACING=Y only at the direction of Informatica Global Customer Support.

Running the PWXUCRGP Utility

You can run a PWXUCRGP utility command from a command line on a Linux, UNIX, or Windows system where PowerExchange is installed.

Make sure that you run the utility under a user ID that has access to the PowerExchange Listener that accesses the CCT file to be read.

To run the utility, navigate to the directory where the PWXUCRGP executable is located. By default, this directory is in the root PowerExchange installation directory. Then enter `PWXUCRGP` followed by the relevant parameters. Use the following syntax:

```
C:\Informatica\PowerExchange\v.r.m PWXUCRGP OUTPUT_FILE=my_file.txt LOCATION=node1
UID=user EPWD=epwd REPORT_SEQUENCE=TABLE REPORT_LEVEL=SUMMARY
DBTYPE=* INSTANCE=* STATUS=* CRNAME=*
```

PWXUCRGP Report Levels of Detail

The PWXUCRGP utility can generate the capture registration report from the CCT file in alternative formats with varying level of detail.

You can specify whether the utility produces a single line of information, summary information, or more detailed column-level information for each capture registration.

Single-Line Report Content

Use the `REPORT_LEVEL=SINGLELINE` parameter setting to print a single line of information for each capture registration that is included in the report.

This report format contains only basic information about the capture registrations. The capture registration information is represented in the report as follows:

```
CRNAME=registration_name STATUS=status CNDISOPT=condense_option TAG=registration_tag
TABLE=schema.table_name
```

The report records are sorted according to the value specified in the `REPORT_SEQUENCE` parameter.

Each single-line entry for a registration in the report includes the following fields:

Field	Description
CRNAME	The user-defined name of the capture registration.
STATUS	The status of the capture registration, which can be one of the following values: <ul style="list-style-type: none">- A: Active- H: Historical- I: Inactive- S: Suspended
CNDISOPT	The Condense option that is set in the capture registration, which can be one of the following values: <ul style="list-style-type: none">- Full- Part- None

Field	Description
TAG	The registration tag name.
TABLE	The source table for which the capture registration was created. This value is in the format <i>schema.table_name</i> .

The following example report entry is for the active capture registration named "emp1" for the Oracle table HDRA.EMP:

```
DBTYPE=ORA          INSTANCE=NA123DTL
=====
CRNAME=emp1          STATUS=A  CNDSOPT=Part TAG=ORANA123DTLemp11    TABLE=HRDA.EMP
```

Summary Report Content

Use the REPORT_LEVEL=SUMMARY parameter setting to print multiple lines of summary information for each capture registration. The report also includes the same information as the SINGLELINE option.

This report format provides the following information for each capture registration:

```
CRNAME=registration_name STATUS=status CNDSOPT=condense_option TAG=registration_tag
TABLE=schema.table_name
User Name: 'user'
Last Updated: 'date'   VRM: 'version'   Edit Sequence: 'edit_seqno'
Registration Type: 'type' Data Columns: 'number'
```

The report records are sorted according to the value specified in the REPORT_SEQUENCE parameter.

Each summary entry for a registration in the report includes the following fields:

Field	Description
CRNAME	The user-defined name of the capture registration.
STATUS	The status of the capture registration, which can be one of the following values: - A : Active - H : Historical - I : Inactive - S : Suspended
CNDSOPT	The Condense option in effect that is set in the capture registration, which can be one of the following values: - Full - Part - None
TAG	The capture registration tag name.
TABLE	The source table name for which the capture registration was created. This value is in the format <i>schema.table_name</i> .
User name	The user name that is specified for the registration group that contains the registration. This field is printed only for the first registration entry and for any subsequent entry in which the user name changed.
Last Updated	The timestamp for the most recent change to the capture registration.

Field	Description
VRM	The PowerExchange version.release.modification level in which the capture registration was most recently updated.
Edit Sequence	The value of the edit sequence (edit_seqno) field for the capture registration. This value is incremented each time the capture registration is edited and prevents concurrent updates to the registration by different users.
Registration Type	Whether the registration is used for log-based or synchronous CDC.
Data Columns	The total number of data columns in each row of the registered table, as recorded in the CCT file.

The report can also contain the following information, depending on the database type:

Database Type	Summary Report Information
ADA	DBID FILENO
DCM	Database ID Table ID Table Name Number of Elements Record Length Recovery Flag
AS4	File Name, if different than table name Journal Library Journal Name Receiver Library Receiver Name Start Position DBDID OBID
DB2	DBID OBID if nonzero Note: These fields are zero if the capture registration was added earlier than DB2 Version 8.
IDM	Subschema Base Record Name Record ID Area Name Page Group Key Radix Compression Indicator Variable Indicator

Database Type	Summary Report Information
IMS	IMS Version Database Name IMS Database Type System Type Segment Name Segment Level Segment Code Parent Code Spans Segments, if present Key Starting Position Key Length DBD Dataset Name, if present Collector Start Time, if present
VSM	Dataset Name

The following example report entry is for the active capture registration named "emp1" for the Oracle table HRDA.EMP:

```

DBTYPE=ORA          INSTANCE=NA123DTL
=====

CRNAME=emp1          STATUS=A  CNDSOPT=Part TAG=ORANA123DTLemp11      TABLE=HRDA.EMP
-----
User Name: 'user1'
Last Updated: '20161208122155' VRM: 'V10.2.0' Edit Sequence: '1'
Registration Type: 'SYNCHRONOUS' Data Columns: '8'

```

Column-Level Report Content

Use the `REPORT_LEVEL=COLUMN` parameter setting to print detailed column information for each capture registration in the report. The report also includes the information that is in the single-line and summary reports.

This report provides the following detailed, column-level information for each capture registration:

```

CRNAME=registration_name STATUS=status CNDSOPT=condense_option TAG=registration_tag
TABLE=schema.table_name
User Name: 'user'
Last Updated: 'date' VRM: 'version' Edit Sequence: 'edit_seqno'
Registration Type: 'type' Data Columns: 'number'

Column Name.           Type.           Precision  Scale CPN Nulls Key Seq
col1                type           precision scale    Y|N Y|N sequence
col2                type           precision scale    Y|N Y|N sequence

```

The report is sorted according to the value specified in the `REPORT_SEQUENCE` parameter.

The report record includes the information shown in the following table.

Item	Description
CRNAME	The user-defined name of the capture registration.
STATUS	The status of the capture registration, which can be one of the following values: <ul style="list-style-type: none"> - A: Active - H: Historical - I: Inactive - S: Suspended
CNDSOPT	The Condense option that is set in the capture registration, which can be one of the following values:: <ul style="list-style-type: none"> - Full - Part - None
TAG	The capture registration tag name.
TABLE	The source table name for which the capture registration was created. This value is in the format <i>schema.table_name</i>
User name	The user name that is specified for the registration group that contains the registration. This field is printed only for the first registration entry and for any subsequent entry in which the user name changed.
Last Updated	The timestamp for the most recent change to the capture registration.
Version	The PowerExchange version.release.modification level in which the capture registration was most recently updated.
Edit Sequence	The value of the edit sequence (edit_seqno) field for the capture registration. This value is incremented each time the capture registration is edited and prevents concurrent updates to the registration by different users.
Registration Type	Whether the registration is used for log-based or synchronous CDC.
Data Columns	The total number of data columns in each row of the registered table, as recorded in the CCT file.
Column Name	The column name.
Type	The column data type.
Precision	The precision of the column, if applicable.
Scale	The scale of the column, if applicable.
CPN	The column code page number, if applicable.
Nulls	Whether the column accepts null values. Valid values: Y or N.
Key	Whether the column is a key column. Valid values: Y or N.
Sequence	For key columns, the column sequence, if available.
Offset	For key columns, the column offset, if available.

Item	Description
Length	For key columns, the key length, if available.
Key Name	For key columns, the key name, if available.

The report can also contain the following information, depending on the database type:

Database Type	Summary Report Information
ADA	DBID FILENO
DCM	Database ID Table ID Table Name Number of Elements Record Length Recovery Flag
AS4	File Name, if different than table name Journal Library Journal Name Receiver Library Receiver Name Start Position DBDID OBID
DB2	DBID OBID if nonzero Note: These fields are zero if the capture registration was added earlier than DB2 Version 8.
IDM	Subschema Base Record Name Record ID Area Name Page Group Key Radix Compression Indicator Variable Indicator

Database Type	Summary Report Information
IMS	IMS Version Database Name IMS Database Type System Type Segment Name Segment Level Segment Code Parent Code Spans Segments, if present Key Starting Position Key Length DBD Dataset Name, if present Collector Start Time, if present
VSM	Dataset Name

The following example report entry is for a capture registration named "alltype1" for the Oracle table dbo.alltypes:

```

DBTYPE=MSS          INSTANCE=QA02CYR
=====

CRNAME=alltype1    STATUS=A  CNDSOPT=Part TAG=MSS02CYRalltype11  TABLE=dbo.alltypes
-----
User Name: 'stuser'
Last Updated: '20170505095940'  VRM: 'V10.1.1'  Edit Sequence: '2'
Registration Type: 'SYNCHRONOUS'  Data Columns: '4'

Column Name          Type          Precision  Scale  CPN  Nulls  Key  Seq
ROW_ID               char           4          0      N    Y     1
tbit                 bit            1          0      N    N
ttinyint             tinyint        3          0      N    N
tsmallint            smallint       5          0      N    N

```

Examples of PWXUCGRP Reports

The following examples show the different types of reports that the PWXUCGRP utility can produce.

Example 1. Single-Line Report

When you run the PWXUCGRP utility with the REPORT_LEVEL=SINGLELINE parameter setting, the report displays a single line of information for each capture registration.

The following command can be used to produce the example single-line report:

```

C:\Informatica\PowerExchangev.r.m PWXUCGRP OUTPUT_FILE=.\Files\LOCAL
\SingleLine_TABLE.txt LOCATION=local UID=user_name PWD=password REPORT_SEQUENCE=TABLE
REPORT_LEVEL=SINGLELINE DBTYPE=* INSTANCE=* STATUS=* CRNAME=*

```

The following example report shows the SINGLELINE output, sorted by table name with no filters applied:

```

2018-04-12 11:49:30                      PWXUCRGP REPORT                      LOCAL
=====

Output file      : .\Files\LOCAL\SingleLine_TABLE.txt
Location        : LOCAL

Report content
-----
Report Sequence  : TABLE
Report Level     : SINGLELINE

Filters for selecting Registrations
-----
Database type    : *
Instance         : *
Status          : *
CR Name         : *

Registration Counts Summary
=====

DB Type   Instance   Total | Active | Part | None
-----
IDL       LOGSID     1 |      1 |    1 |    0
MSS       CYRI000     6 |      6 |    6 |    0
ORA       EL111DTL    1 |      1 |    1 |    0
UDB       JAPAN943    4 |      4 |    4 |    0
All       *all*      22 |     22 |   22 |    0
-----

DBTYPE=IDL      INSTANCE=LOGSID
=====
CRNAME=aa        STATUS=A  CNDSOPT=Part TAG=IDLLOGSIDaa1      TABLE=aaempss01.map1_INSURANCE_PLAN

DBTYPE=MSS       INSTANCE=CYRI000
=====
CRNAME=char1     STATUS=A  CNDSOPT=Part TAG=MSSCYRI000char11  TABLE=dbo.testChar
CRNAME=nchar1    STATUS=A  CNDSOPT=Part TAG=MSSCYRI000nchar11  TABLE=dbo.testNChar
CRNAME=nvarchar  STATUS=A  CNDSOPT=Part TAG=MSSCYRI000nvarchar1  TABLE=dbo.testNVarChar
CRNAME=ncchar1   STATUS=A  CNDSOPT=Part TAG=MSSCYRI000ncchar11  TABLE=dbo.testNchar
CRNAME=text1     STATUS=A  CNDSOPT=Part TAG=MSSCYRI000text11  TABLE=dbo.testText
CRNAME=varchar1  STATUS=A  CNDSOPT=Part TAG=MSSCYRI000varchar11  TABLE=dbo.testVarChar

DBTYPE=ORA       INSTANCE=EL111DTL
=====
CRNAME=empl      STATUS=A  CNDSOPT=Part TAG=ORAE111DTLempl1  TABLE=HRDA.EMP

DBTYPE=UDB       INSTANCE=JAPAN943
=====
CRNAME=table2    STATUS=A  CNDSOPT=Part TAG=UDBJAPAN943table21  TABLE=MYDB.TABLE2
CRNAME=table4    STATUS=A  CNDSOPT=Part TAG=UDBJAPAN943table41  TABLE=MYDB.TABLE4

```

Example 2. Summary Report

When you run the PWXUCRGP utility with the REPORT_LEVEL=SUMMARY parameter setting, the report displays summary information for each capture registration.

The following command can be used to produce the example summary report:

```

C:\Informatica\PowerExchange\v.r.m PWXUCRGP OUTPUT_FILE=.\Files\LOCAL\Summary_TABLE.txt
LOCATION=local UID=user_name PWD=password REPORT_SEQUENCE=TABLE REPORT_LEVEL=SUMMARY
DBTYPE=* INSTANCE=* STATUS=* CRNAME=*

```

The following example report shows the SUMMARY format, sorted by table name and with no filtering applied:

```

2017-12-12 11:49:31                      PWXUCRGP REPORT                      LOCAL
=====

Output file      : .\Files\LOCAL\Summary_TABLE.txt
Location        : LOCAL

Report content

```

```

-----
Report Sequence      : TABLE
Report Level        : SUMMARY

Filters for selecting Registrations
-----
Database type       : *
Instance            : *
Status              : *
CR Name             : *

Registration Counts Summary
=====

```

DB Type	Instance	Total	Active	Part	None
IDL	LOGSID	1	1	1	0
MSS	CYRI000	6	6	6	0
	RB02CYR	1	1	1	0
MSS	*all*	7	7	7	0
ORA	EL111DTL	1	1	1	0
UDB	SAMPLE	3	3	3	0
All	*all*	12	12	12	0

```

-----
DBTYPE=IDL          INSTANCE=LOGSID
=====

CRNAME=aa            STATUS=A CNDSOPT=Part TAG=IDLLOGSIDaa1    TABLE=aaempss01.map1_INSURANCE_PLAN
-----
User Name: 'ABCD'
Last Updated: '20161207173435' VRM: 'V10.2.0' Edit Sequence: '1'
Registration Type: 'LOG-BASED' Data Columns: '13'
IDMS: SUBSCHEMA 'empdemo' BASEREC 'INSURANCE_PLAN' COMPRESS IND 'N' VARIABLE IND 'N'

DBTYPE=MSS          INSTANCE=CYRI000
=====

CRNAME=char1         STATUS=A CNDSOPT=Part TAG=MSSCYRI000char11 TABLE=dbo.testChar
-----
Last Updated: '20101104113518' VRM: 'V9.1.0' Edit Sequence: '3'
Registration Type: 'SYNCHRONOUS' Data Columns: '3'

CRNAME=ntext1        STATUS=A CNDSOPT=Part TAG=MSSCYRI000ntext11 TABLE=dbo.testNText
-----
Last Updated: '20101104134524' VRM: 'V9.1.0' Edit Sequence: '3'
Registration Type: 'SYNCHRONOUS' Data Columns: '3'

CRNAME=nvarchar      STATUS=A CNDSOPT=Part TAG=MSSCYRI000nvarchar1 TABLE=dbo.NVarchar
-----
Last Updated: '20101104134454' VRM: 'V9.1.0' Edit Sequence: '3'
Registration Type: 'SYNCHRONOUS' Data Columns: '3'

CRNAME=nchar1        STATUS=A CNDSOPT=Part TAG=MSSCYRI000nchar11 TABLE=dbo.Nchar
-----
Last Updated: '20101104134410' VRM: 'V9.1.0' Edit Sequence: '3'
Registration Type: 'SYNCHRONOUS' Data Columns: '3'

CRNAME=text1         STATUS=A CNDSOPT=Part TAG=MSSCYRI000text11 TABLE=dbo.Text
-----
Last Updated: '20101104160450' VRM: 'V9.1.0' Edit Sequence: '4'
Registration Type: 'SYNCHRONOUS' Data Columns: '3'

CRNAME=varchar1      STATUS=A CNDSOPT=Part TAG=MSSCYRI000varchar11 TABLE=dbo.Varchar
-----
Last Updated: '20101104113749' VRM: 'V9.1.0' Edit Sequence: '3'
Registration Type: 'SYNCHRONOUS' Data Columns: '3'

DBTYPE=MSS          INSTANCE=RB02CYR
=====

CRNAME=alltype1      STATUS=A CNDSOPT=Part TAG=MSSRB02CYRalltype11 TABLE=dbo.alltypes
-----
User Name: 'hrda'
Last Updated: '20170505095940' VRM: 'V10.1.1' Edit Sequence: '2'
Registration Type: 'SYNCHRONOUS' Data Columns: '27'

DBTYPE=ORA          INSTANCE=EL111DTL
=====

CRNAME=empl          STATUS=A CNDSOPT=Part TAG=ORAE111DTLempl11 TABLE=HRDA.EMP

```

```

-----
User Name: 'hrda'
Last Updated: '20161208122155' VRM: 'V10.2.0' Edit Sequence: '1'
Registration Type: 'SYNCHRONOUS' Data Columns: '8'

DBTYPE=UDB INSTANCE=SAMPLE
=====

CRNAME=rfact STATUS=A CNDSOPT=Part TAG=UDBSAMPLErfact1 TABLE=MYDB.ACT
-----
User Name: 'MYDB'
Last Updated: '20161205121009' VRM: 'V10.2.0' Edit Sequence: '1'
Registration Type: 'SYNCHRONOUS' Data Columns: '3'

CRNAME=adep STATUS=A CNDSOPT=Part TAG=UDBSAMPLEadep1 TABLE=MYDB.ADEFUSR
-----
Last Updated: '20161207093112' VRM: 'V10.2.0' Edit Sequence: '1'
Registration Type: 'SYNCHRONOUS' Data Columns: '2'

CRNAME=mydb0024 STATUS=A CNDSOPT=Part TAG=UDBSAMPLEmydb00241 TABLE=MYDB.BIN_INT_SRC
-----
Last Updated: '20161205121009' VRM: 'V10.2.0' Edit Sequence: '1'
Registration Type: 'SYNCHRONOUS' Data Columns: '2'

```

Example 3. Column-Level Report

When you run the PWXUCGRP utility with the REPORT_LEVEL=COLUMNS parameter setting, the report displays detailed column-level information for each capture registration.

The following command can be used to produce the example column-level report:

```

C:\Informatica\PowerExchangev.r.m PWXUCGRP OUTPUT_FILE=.\Files\LOCAL\Columns_CRNAME.txt
LOCATION=MHZEUSER2 UID=user_name PWD=password REPORT_SEQUENCE=CRNAME
REPORT_LEVEL=COLUMNS DBTYPE=* INSTANCE=* STATUS=* CRNAME=*

```

The following example report shows the COLUMNS output, sorted by CRNAME with no filters applied:

```

2018-01-15 09:13:01 PWXUCGRP REPORT MHZEUSER2
=====

Output file      : .\Files\MHZEUSER2\Columns_CRNAME.txt
Location        : MHZEUSER2

Report content
-----
Report Sequence  : CRNAME
Report Level     : COLUMNS

Filters for selecting Registrations
-----
Database type    : *
Instance         : *
Status          : *
CR Name         : *

Registration Counts Summary
=====

DB Type  Instance  Total | Active | Part | None | Full
-----
ADA      ADA8242   1 | 1 | 1 | 0 | 0
DCM      MUFR15Q0   1 | 1 | 1 | 0 | 0
IDL      RVFLGSID   5 | 2 | 2 | 0 | 1
          LOGSID    1 | 1 | 1 | 0 |
IDL      *all*     6 | 6 | 3 | 3 | 0
IMS      IMS9      6 | 5 | 2 | 1 | 3
VSM      FLDVSAM    1 | 1 | 1 | 0 | 0
-----
All      *all*     15 | 11 | 8 | 1 | 4
-----

DBTYPE=ADA INSTANCE=ADA8242
=====

CRNAME=ada8242 STATUS=A CNDSOPT=Part TAG=ADAADA8242DBID08242FILEID00351 TABLE=ada8242.map351a_ADA_RECORD

```

```
-----
Last Updated: '20141218144631' Version: 'V9.6.1' Edit Sequence: '2'
Registration Type: 'LOG-BASED' Data Columns: '9'
Adabas: DB Id: '8242' File Number: '351'
```

Column Name.	Type.	Precision	Scale	CPN	Nulls	Key	Seq	Offset	Length
AA_field	CHAR	8	0			Y	N		
AC_field	CHAR	20	0			Y	N		
AE_field	CHAR	20	0			Y	N		
AF_field	CHAR	1	0			Y	N		
AG_field	CHAR	1	0			Y	N		
AH_field	PACKED	11	0			Y	N		
AI_field	CHAR	20	0			Y	N		
AJ_field	CHAR	20	0			Y	N		
AN_field_10	VARCHAR	1024	0			Y	N		

```
DBTYPE=DCM      INSTANCE=MUFR15Q0
=====
```

```
CRNAME=aa      STATUS=A  CNDSOPT=Part TAG=DCMMUFR15Q0DBID00001STNPAYV1      TABLE=aa.pay_PAYROLL
-----
```

```
User name: 'USER2'
Last Updated: '20161207151954' Version: 'V10.2.0' Edit Sequence: '1'
Registration Type: 'SYNCHRONOUS' Data Columns: '7'
Datacom: DB Id: '00001' Table Id: '1' Table Name: 'PAY' Elements: '6' Record Length: '40' Recovery Flag: 'Y'
```

Element Name	Offset	Length
PYIDT	1	7
FIGS	8	32
PAYRC	1	39

Column Name.	Type.	Precision	Scale	CPN	Nulls	Key	Seq	Offset	Length	Key Name.
NUMBER	UZONED	5	0			N	Y	1	1	5 NUMBER
ACTIVITY_CODE	CHAR	1	0			N	N			
ACTIVITY_STATUS	CHAR	1	0			N	N			
CURRENT_RATE	UZONED	8	2			N	N			
YTD_WAGES	UZONED	8	2			N	N			
YTD_COMMISSION	UZONED	8	2			N	N			
YTD_TAX	UZONED	8	2			N	N			

```
DBTYPE=IDL      INSTANCE=RVFLGSID
=====
```

```
CRNAME=empss01 STATUS=A  CNDSOPT=Part TAG=IDLRVFLGSIDempss011      TABLE=empss01.map1_employee
-----
```

```
User name: 'USER2'
Last Updated: '20171211170904' Version: 'V10.2.0' Edit Sequence: '1'
Registration Type: 'LOG-BASED' Data Columns: '9'
IDMS: Subschema: 'empdemo' Base Record Name: 'EMPLOYEE' Id: '415'
IDMS: Area Name: 'EMP-DEMO-REGION' Page Group: '0' Radix: '8'
```

Column Name.	Type.	Precision	Scale	CPN	Nulls	Key	Seq	Offset	Length
EMP_ID_0415	UZONED	4	0			N	N		
EMP_FIRST_NAME_0415	CHAR	10	0			N	N		
EMP_LAST_NAME_0415	CHAR	15	0			N	N		
EMP_STREET_0415	CHAR	20	0			N	N		
EMP_CITY_0415	CHAR	15	0			N	N		
EMP_STATE_0415	CHAR	2	0			N	N		
EMP_ZIP_FIRST_FIVE_0415	CHAR	5	0			N	N		
EMP_ZIP_LAST_FOUR_0415	CHAR	4	0			N	N		
EMP_PHONE_0415	UZONED	10	0			N	N		

```
CRNAME=idmsqa STATUS=A  CNDSOPT=Part TAG=IDLRVFLGSIDidmsqa1      TABLE=idmsqa.stcrss01_student
-----
```

```
Last Updated: '20171211171039' Version: 'V10.2.0' Edit Sequence: '1'
Registration Type: 'LOG-BASED' Data Columns: '6'
IDMS: Subschema: 'stcrseg' Base Record Name: 'STUD' Id: '0'
IDMS: Area Name: 'STUDCRSE-REGION' Page Group: '0' Radix: '8'
```

Column Name.	Type.	Precision	Scale	CPN	Nulls	Key	Seq	Offset	Length
STUD_NAME	CHAR	40	0			N	N		
STUD_ADDRESS_1	CHAR	40	0			N	N		
STUD_ADDRESS_2	CHAR	40	0			N	N		
STUD_CITY	CHAR	30	0			N	N		
STUD_STATE	CHAR	2	0			N	N		
STUD_ZIP	CHAR	9	0			N	N		

```
DBTYPE=IMS      INSTANCE=IMS9
=====
```

```
CRNAME=dtld0006 STATUS=A  CNDSOPT=Full TAG=IMLIMS9dtld0006100000      TABLE=ims9.dtld0006.COMPLETE_HIERARCHY
-----
```

```
User name: 'USER2'
Last Updated: '2017121114641' Version: 'V10.2.0' Edit Sequence: '1'
Registration Type: 'LOG-BASED' Data Columns: '6'
Data Map: 'dtld0006'
```

IMS: Version: '9.10' DB Name: 'DTLD006' DB Type: 'HIDAM VSAM' System Type: 'Y'
 IMS: Segment Name: 'STUDENT' Level: '1' Code: '1' Parent Code: '0'
 IMS: Spans Segments: 'Y'
 IMS: Key Start '161' Key Length '12'

Column Name.	Type.	Precision	Scale	CPN	Nulls	Key	Seq
STUDENT:PNAME	CHAR	40	0		N	N	
STUDENT:ADDRESS1	CHAR	40	0		N	N	
STUDENT:ADDRESS2	CHAR	40	0		N	N	
STUDENT:CITY	CHAR	30	0		N	N	
STUDENT:STATE	CHAR	2	0		N	N	
STUDENT:ZIP	CHAR	9	0		N	N	

CRNAME=dtld0008 STATUS=I CNDISOPT=Part TAG=IMLIMS9dtld0008100000 TABLE=ims9.dtld008.COMPLETE_HIERARCHY

 Last Updated: '2017121114734' Version: 'V10.2.0' Edit Sequence: '1'
 Registration Type: 'LOG-BASED' Data Columns: '9'
 Data Map: 'dtld008'
 IMS: Version: '9.10' DB Name: 'DTLD008' DB Type: 'HDAM OSAM' System Type: 'Y'
 IMS: Segment Name: 'NAMEMAST' Level: '1' Code: '1' Parent Code: '0'
 IMS: Spans Segments: 'Y'
 IMS: Key Start '0' Key Length '60'

Column Name.	Type.	Precision	Scale	CPN	Nulls	Key	Seq
NAMEMAST:EMPLOYEE	CHAR	60	0		N	N	
NAMEMAST:MANNBR	CHAR	15	0		N	N	
NAMEMAST:ADDR	CHAR	75	0		N	N	
NAMESKIL:TYPE	CHAR	21	0		Y	N	
NAMESKIL:STDLEVL	CHAR	20	0		Y	N	
ADDRESS:HOMEADDR	CHAR	100	0		Y	N	
ADDRESS:COMAILOC	CHAR	100	0		Y	N	
PAYROLL:BASICPAY	PACKED	29	0		Y	N	
PAYROLL:HOURS	PACKED	29	0		Y	N	

CRNAME=dtld0009 STATUS=I CNDISOPT=None TAG=IMLIMS9dtld0009100000 TABLE=ims9.dtld009.COMPLETE_HIERARCHY

 Last Updated: '2017121114831' Version: 'V10.2.0' Edit Sequence: '1'
 Registration Type: 'LOG-BASED' Data Columns: '8'
 Data Map: 'dtld009'
 IMS: Version: '9.10' DB Name: 'DTLD009' DB Type: 'HDAM OSAM' System Type: 'Y'
 IMS: Segment Name: 'SKILMAST' Level: '1' Code: '1' Parent Code: '0'
 IMS: Spans Segments: 'Y'
 IMS: Key Start '0' Key Length '21'

Column Name.	Type.	Precision	Scale	CPN	Nulls	Key	Seq
SKILMAST:TYPE	CHAR	21	0		N	N	
SKILMAST:STDCODE	CHAR	10	0		N	N	
SKILNAME:EMPLOYEE	CHAR	60	0		Y	N	
SKILNAME:STDLEVL	CHAR	20	0		Y	N	
EXPR:PREVJOB	CHAR	10	0		Y	N	
EXPR:CLASSIF	CHAR	10	0		Y	N	
EDUC:GRADLEVL	CHAR	10	0		Y	N	
EDUC:SCHOOL	CHAR	65	0		Y	N	

DBTYPE=VSM INSTANCE=FLDVSAM
 =====

CRNAME=cdcfld1 STATUS=A CNDISOPT=Part TAG=VSAMUSER2.CDCFIELD.KSDS1 TABLE=mvs800.cdcfield1_t1

 Last Updated: '20121105130114' Version: 'V9.5.1' Edit Sequence: '1'
 Registration Type: 'SYNCHRONOUS' Data Columns: '3'
 Registration Location: 'mhzDefgh'
 Vsam: Dataset Name: 'USER2.CDCFIELD.KSDS1'

Column Name.	Type.	Precision	Scale	CPN	Nulls	Key	Seq	Offset	Length
F1	CHAR	4	0		N	N			
F2	CHAR	20	0		N	N			
F3	CHAR	10	0		N	N			

CHAPTER 23

PWXUDMX - Data Maps Update Time ECSA Memory Utility

This chapter includes the following topics:

- [PWXUDMX Utility Overview, 296](#)
- [Supported Operating Systems for the PWXUDMX Utility, 296](#)
- [Running the PWXUDMX Utility on z/OS, 297](#)
- [PWXUDMX Commands and Parameters, 297](#)

PWXUDMX Utility Overview

Use the PWXUDMX utility to allocate, display, and delete ECSA memory, which holds time stamps of the latest updates to data maps files, and to modify the use counts of a file.

This processing is relevant if you configure data maps caching in multiple jobs mode by defining DMXCACHE_MULTIPLEJOBS=Y in the DBMOVER configuration file.

With the PWXUDMX utility, you can complete the following tasks:

- Allocate less than the 4096 bytes of ECSA memory that the system dynamically allocates.
- Delete ECSA memory.
- Display the contents of ECSA memory with file names and time stamps in legible format.
- Display the contents of ECSA memory in hexadecimal format.
- If a PowerExchange Listener or netport job does not shut down cleanly, decrement the use count of a file.
- Increment the use count of a file.

Supported Operating Systems for the PWXUDMX Utility

The PWXUDMX utility runs on z/OS systems.

Running the PWXUDMX Utility on z/OS

You run the PWXUDMX utility by submitting the PWXUDMX job. The input control statements for this utility are read from SYSIN.

The following is an example of JCL to use when you run this utility on z/OS.

```
//jobname JOB 'UDMX',MSGLEVEL=(1,1),MSGCLASS=X,
// CLASS=A,NOTIFY=&SYSUID
//*
//STEP1 EXEC PGM=PWXUDMX,
// PARM='CMD=command'
//STEPLIB DD DSN=CEE.SCEERUN,DISP=SHR
// DD DSN=&HLQ..LOADLIB,DISP=SHR
//DATAMAP DD DSN=&HLQ..V1.DATAMAPS,DISP=SHR
//DTLMSG DD DSN=&HLQ..DTLMSG,DISP=SHR
//DTLCFG DD DSN=&HLQ..RUNLIB(DBMOVER),DISP=SHR
//DTLKEY DD DSN=&HLQ..RUNLIB(LICENSE),DISP=SHR
//DTLSGN DD DSN=&HLQ..RUNLIB(SIGNON),DISP=SHR
//DTLOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//DTLLOG DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//CEEDUMP DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DTLCFG DD *
```

The JCL statements are:

JOB

Defines the PWXUDMX job card to z/OS, including the job name.

EXEC PGM=PWXUDMX

Identifies the name of the program, PWXUDMX, to run.

PARM='CMD=command'

Identifies the name of the PWXUDMX command to run. For a description of the commands, see [“PWXUDMX Commands and Parameters” on page 297](#).

STEPLIB DD

Defines the PowerExchange LOAD library that contains the utility.

DATAMAP DD

If you do not specify the optional FILE parameter for the DECREMENT_FILE_COUNT or INCREMENT_FILE_COUNT command, PowerExchange modifies the file count of the file specified in the DATAMAP DD statement.

SYSPRINT DD

Defines the print location for the output of the DISPLAY_ECSCA or DUMP_ECSCA commands.

PWXUDMX Commands and Parameters

This section describes the commands that you can enter in the CMD statement of the PWXUDMX syntax and the command-specific parameters.

CREATE_ECSA Command

Creates ECSA memory.

```
//STEP1 EXEC PGM=PWXUDMX,  
//      PARM='CMD=CREATE_ECSA [LENGTH=length]'
```

To allocate less than the 4096 bytes of ECSA memory that the system dynamically allocates, issue this command with the optional LENGTH parameter. For the *length* variable, specify the amount of storage to allocate for ECSA memory. The PWXUDMX utility allocates the specified amount of ECSA memory and creates a named token called PWX_DMXTIME_1, which defines the address of the ECSA memory.

Maximum value is 4096 bytes.

DECREMENT_FILE_COUNT Command

Decrements the use count of a file.

```
//STEP1 EXEC PGM=PWXUDMX,  
//PARM='CMD=DECREMENT_FILE_COUNT [FILE=file]'
```

If a PowerExchange Listener or netport job does not shut down cleanly, issue this command to correct the use count.

If you do not specify the optional FILE parameter, the PWXUDMX utility decrements the use count of the file specified in the DATAMAP DD statement.

DELETE_ECSA Command

Deletes ECSA memory.

```
//STEP1 EXEC PGM=PWXUDMX,  
//      PARM='CMD=DELETE_ECSA [FORCE={N|Y}]'
```

Issue this command to delete ECSA memory when you uninstall PowerExchange.

If ECSA memory has a nonzero use count, use either the DECREMENT_FILE_COUNT command to reduce the use count or the optional FORCE=Y parameter on the DELETE_ECSA command to force deletion of ECSA memory.

When you run the DELETE_ECSA command, the PWXUDMX utility deletes the ECSA memory and deletes the named token called PWX_DMXTIME_1, which defines the address of the ECSA memory.

DISPLAY_ECSA Command

Displays ECSA memory.

```
//STEP1 EXEC PGM=PWXUDMX,  
//      PARM='CMD=DISPLAY_ECSA'
```

Display the contents of ECSA memory, including file names and time stamps in legible format.

DUMP_ECSA Command

Displays ECSA memory in hexadecimal format.

```
//STEP1 EXEC PGM=PWXUDMX,  
//      PARM='CMD=DUMP_ECSA'
```

INCREMENT_FILE_COUNT Command

Increments the use count of a file.

```
//STEP1 EXEC PGM=PWXUDMX,  
//          PARM='CMD=INCREMENT_FILE_COUNT [FILE=file]'
```

If you do not specify the optional FILE parameter, PowerExchange increments the file count of the file specified in the DATAMAP DD statement.

CHAPTER 24

PWXUGSK - SSL Reporting Utility for z/OS

This chapter includes the following topics:

- [PWXUGSK Utility Overview, 300](#)
- [Supported Operating Systems for the PWXUGSK Utility, 300](#)
- [PWXUGSK Control Statement Syntax, 301](#)
- [PWXUGSK Commands and Parameters, 301](#)
- [Running the PWXUGSK Utility, 303](#)
- [PWXUGSK Utility Reports, 304](#)

PWXUGSK Utility Overview

Use the PWXUGSK utility to generate reports about SSL libraries and certificates that were generated on z/OS for the PowerExchange Listener. You can also determine the validity of certificates that are available to a specified user.

With the PWXUGSK utility, you can complete the following tasks:

- Verify that a specified user ID has the authority to view security certificates for the PowerExchange Listener on z/OS, that the certificates are current and valid, and that the AT-TLS rules can intercept inbound requests, remove the TLS information and send TCP/IP packets to the listener.
- Run a certificate report to view certificate information from a RACF keyring or SAF database.
- Run a cipher report to view the cipher suites that are available on the z/OS system.
- Run an error-code report to view all possible system SSL errors. After an SSL failure, particular instances of these errors can be found in the TCP/IP JES message log.

Supported Operating Systems for the PWXUGSK Utility

The PWXUGSK utility runs on z/OS systems.

PWXUGSK Control Statement Syntax

Use the following general syntax to specify control statements for the PWXUGSK utility:

```
PWXUGSK CMD=command_name command-specific parameters
```

PWXUGSK Commands and Parameters

This section describes the commands that you can enter in the CMD statement of the PWXUGSK syntax and the parameters available for each command.

PING Command

Use the PING command to verify that a secure connection can be established between the machine from which you issue the command and a PowerExchange Listener on a remote node.

When you issue the PING command, the PWXUGSK utility steps through each phase of a secure connection to verify that the connection will succeed. You can use the information returned from this command to troubleshoot any issues that might occur.

```
PWXUGSK CMD=PING PING_LOCATION=node_name  
[PING_UID=user_name{PING_PWD=password|PING_EPWD=encrypted_password}]
```

The PING command has the following parameters:

PING_LOCATION=*node_name*

The node name of the remote PowerExchange Listener to ping for the secure connection.

PING_UID=*user_name*

A user name that can be used to establish the secure connection. The user must have the authority to view SSL certificates on the remote node specified by PING_LOCATION.

PING_PWD=*password*

The password associated with the user name specified by PING_UID. Specify either a password or an encrypted password to decrypt the certificate files on the remote node.

PING_EPWD=*encrypted_password*

An encrypted password associated with the user name specified by PING_UID. Specify either a password or an encrypted password to decrypt the certificate files on the remote node.

REPORT_CERTIFICATES Command

The REPORT_CERTIFICATES command generates a report that lists information about all of the security certificates in a keyring or database that are available to the user issuing the command.

The command has the following syntax:

```
PWXUGSK CMD=REPORT CERTIFICATES [LOC_TYPE={KEYRING|DATABASE}] [LOC_NAME=name]  
[DB_PWD=password] [DB_EPWD=encrypted_password] [VERBOSE={N|Y}]
```

The REPORT_CERTIFICATES command has the following parameters:

LOC_TYPE={KEYRING|DATABASE}

The location of the certificates to include in the report. The PWXUGSK utility can produce reports from keyrings or certificate databases on z/OS:

KEYRING

Certificates stored in a keyring on z/OS.

DATABASE

Certificates stored in a database. If you specify DATABASE, you must also include either a database password or encrypted password.

Default is **KEYRING**.

LOC_NAME

The name of the keyring or certificate database for which information is reported.

DB_PWD

If LOC_TYPE=DATABASE, a password that is used to access the database specified by the LOC_NAME parameter. Either the DB_PWD or DB_EPWD parameter must be specified if a certificate database is specified.

DB_EPWD

If LOC_TYPE=DATABASE, an encrypted password that is used to access the database specified by the LOC_NAME parameter. Either the DB_PWD or DB_EPWD parameter must be specified if a certificate database is specified.

VERBOSE={N|Y}

The output format for the report. If you specify VERBOSE=Y, the report includes detailed information for each certificate. If you specify N, the report includes only summary information for each certificate. Default is N.

REPORT_CIPHERS

The REPORT_CIPHERS command generates a report that lists all of the ciphers available on the z/OS system where the PowerExchange Listener runs.

The command has the following syntax:

```
PWXUGSK CMD=REPORT_CIPHERS
```

The REPORT_CIPHERS command has no other parameters,

Note: For PowerExchange Listeners on z/OS machines, the report returns all available ciphers, but AT-TLS rules might limit a user to a subset of the available ciphers.

REPORT_ERROR_CODES Command

The REPORT_ERROR_CODES command generates a report that lists all possible z/OS system SSL error codes, either during the secure connection process or when processing data packets.

The command has the following syntax:

```
PWXUGSK CMD=REPORT_ERROR_CODES
```

The REPORT_ERROR_CODES command has no other parameters.

Running the PWXUGSK Utility

You can run the utility by submitting a PWXUGSK job. The STEPLIB must include the load library that contains the GSKSSL member.

The following example shows JCL to generate a certificates report using the PWXUGSK utility:

```
//GSKPINGN JOB 'GSK          ',MSGLEVEL=(1,1),MSGCLASS=X,
//          CLASS=A,NOTIFY=&SYSUID
//*
//*=====
//* RUN PWXUGSK
//*=====
//STEP1     EXEC PGM=PWXUGSK,REGION=900M,
//          PARM='CMD=REPORT_CERTIFICATES LOC_TYPE=KEYRING
//          VERBOSE=N'
//*
//STEPLIB DD DISP=SHR,DSN=&HLQ..LOADLIB
//          DD DISP=SHR,DSN=&SCERUN
//          DD DISP=SHR,DSN=&RUNLIB
//*
//DTLMSG DD DISP=SHR,DSN=&HLQ..DTLMSG
//DTLKEY DD DISP=SHR,DSN=&RUNLIB (LICENSE)
//DTLSGN DD DISP=SHR,DSN=&RUNLIB (SIGNON)
//DTLLOG DD DUMMY,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=133)
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//CEEDUMP DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//*
//DTLCFG DD *
```

The following example shows JCL to use the PING command to verify a secure connection to a PowerExchange client:

```
//GSKPINGN JOB 'GSK          ',MSGLEVEL=(1,1),MSGCLASS=X,
//          CLASS=A,NOTIFY=&SYSUID
//*
//*=====
//* RUN PWXUGSK
//*=====
//STEP1     EXEC PGM=PWXUGSK,REGION=900M,
//          PARM='CMD=PING PING_LOCATION=ZSY216495
//          PING_UID=MY PING_PWD=MYPWD VERBOSE=N'
//*
//STEPLIB DD DISP=SHR,DSN=&HLQ..LOADLIB
//          DD DISP=SHR,DSN=&SCERUN
//          DD DISP=SHR,DSN=&RUNLIB
//*
//DTLMSG DD DISP=SHR,DSN=&HLQ..DTLMSG
//DTLKEY DD DISP=SHR,DSN=&RUNLIB (LICENSE)
//DTLSGN DD DISP=SHR,DSN=&RUNLIB (SIGNON)
//DTLLOG DD DUMMY,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=133)
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//CEEDUMP DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//*
//DTLCFG DD *
```

PWXUGSK Utility Reports

This section describes the reports that you can generate with the PWXUGSK utility.

Certificates Report

The certificates report provides certificate information from a RACF keyring or SAF database.

To generate a certificate report, enter the following command:

```
PWXUGSK CMD=REPORT_CERTIFICATES [LOC_TYPE={KEYRING|DATABASE}] [VERBOSE={N|Y}]
```

The following output is an example of a certificate report with the VERBOSE option set to N:

```
PWXUGSK REPORT_CERTIFICATES
=====
Verbose N
-----
Processing console program. PWXUGSK CMD=REPORT_CERTIFICATES VERBOSE=N LOC_TYPE=KEYRING
LOC_NAME=ATTLS_keyring
VERBOSE=''

Report for command REPORT_CERTIFICATES
=====

Labels
-----
CA Certificates without Private keys . labels=1:  LOCALCA
Subject Certificates with Private keys. labels=1:  STQACert1

Certificate 1 Label 'LOCALCA'
-----
Record Type           = 2 (CA Certificate)
Algorithm type        = 22 (sha1WithRsaEncryption)
Version               = 2
Serial Number         = 00
Record Id             = 1
Issuer Record Id      = 1
Not Before Timestamp  = 2006-11-06 00:00:00
Not After Timestamp   = 2040-12-31 23:59:59

Subject
CN = irrcerta.mhz8902.informatica.com OU = zOS.Admin O = Informatica C = GB
Issuer
CN = irrcerta.mhz8902.informatica.com OU = zOS.Admin O = Informatica C = GB
*** Self-signed certificate
*** SITE certificate with user id 'IRRCERTA'

Certificate validated OK by gsk_validate_certificate_mode

Certificate 2 Label 'STQACert1'
-----
Record Type           = 3 (Subject Certificate & Private Key)
Algorithm type        = 22 (sha1WithRsaEncryption)
Version               = 2
Serial Number         = 0D
Record Id             = 2
Issuer Record Id      = 1
Not Before Timestamp  = 2008-01-15 00:00:00
Not After Timestamp   = 2031-12-31 23:59:59

Subject
CN = STQAtest.mhz890-2.informatica.com OU = zOS.Admin O = Informatica C = GB
Issuer
CN = irrcerta.mhz8902.informatica.com OU = zOS.Admin O = Informatica C = GB

Certificate validated OK by gsk_validate_certificate_mode
Issuer_record_id = 1
```



```
CA certificates      = 1
Subject certificates = 1
Total certificates   = 2
```

The following output is an example of a certificate report with the VERBOSE option set to Y:

```
PWXUGSK REPORT_CERTIFICATES
=====
Verbose Y
-----
Processing console program. PWXUGSK CMD=REPORT_CERTIFICATES VERBOSE=Y LOC_TYPE=KEYRING
LOC NAME=ATTLS_keyring
VERBOSE='Y'

Report for command REPORT_CERTIFICATES
=====

Initializing
-----
Calling gsk_get_cms_vector()
Mask = 2047_X'000007FF' (API: LVL1, LVL2, LVL3, LVL4, LVL5, LVL6, LVL7, LVL8, LVL9,
LVL10, LVL11)
Number of functions = 223
gsk_cms_vector. Returned size=892. Compiled size=636

Calling gsk_open_keyring(). Name='ATTLS_keyring'. Function address=1F173BF0
gsk_open_keyring() returned db_handle=1EF7A610 num_records=2

Labels
-----
gsk_get_record_labels()CA Certificates without Private keys labels=1 LOCALCA
gsk_get_record_labels()Subject Certificates with Private keys labels=1 STQACert1

Looping calling gsk_get_record_by_index() to get certificates

Certificate 1 Label 'LOCALCA'
-----
Record Type           = 2 (CA Certificate)
Algorithm type        = 22 (shaWithRsaEncryption)
Algorithm oid count   = 7
Version               = 2
Serial Number         = 00
Record Id             = 1
Issuer Record Id      = 1
Not Before Timestamp  = 2006-11-06 00:00:00
Not After Timestamp   = 2040-12-31 23:59:59
Subject
Name Type = 1 (DN). Rdn attributes=4
attributeType = 13 (countryName)          length= 2 data='GB'
attributeType =  9 (organizationName)     length=11 data='Informatica'
attributeType = 10 (organizationalUnitName) length= 9 data='zOS.Admin'
attributeType =  6 (commonName)           length=32
data='irrcerta.mhz8902.informatica.com'
CN = irrcerta.mhz8902.informatica.com OU = zOS.Admin O = Informatica C = GB

Issuer
Name Type = 1 (DN). Rdn attributes=4
attributeType = 13 (countryName)          length= 2 data='GB'
attributeType =  9 (organizationName)     length=11 data='Informatica'
attributeType = 10 (organizationalUnitName) length= 9 data='zOS.Admin'
attributeType =  6 (commonName)           length=32
data='irrcerta.mhz8902.informatica.com'
CN = irrcerta.mhz8902.informatica.com OU = zOS.Admin O = Informatica C = GB

*** Self-signed certificate
*** SITE certificate with user id 'IRRCERTA'

Subject Name Attributes
Buffer: length=100
```

```
data=3062310B300906035504061302474231143012060355040A130B496E666F726D61746963613112301006
0355040B13097A4F532E4164
6D696E312930270603550403132069727263657274612E6D687A383930322E696E666F726D61746963612E636
F6D
```

```
C = GB
O = Informatica
OU = zOS.Adminca
CN = irrcerta.mhz8902.informatica.com
```

Issuer Name Attributes

Buffer: length=100

```
data=3062310B300906035504061302474231143012060355040A130B496E666F726D61746963613112301006
0355040B13097A4F532E4164
6D696E312930270603550403132069727263657274612E6D687A383930322E696E666F726D61746963612E636
F6D
```

```
C = GB
O = Informatica
OU = zOS.Adminca
CN = irrcerta.mhz8902.informatica.com
```

F51 ValidateCertMode starts

Calling gsk_validate_certificate_mode

Certificate validated OK by gsk_validate_certificate_mode

Certificate 2 Label 'STQACert1'

```
-----
Record Type           = 3 (Subject Certificate & Private Key)
Algorithm type        = 22 (sha1WithRsaEncryption)
Algorithm oid count   = 7
Version               = 2
Serial Number         = 0D
Record Id             = 2
Issuer Record Id      = 1
Not Before Timestamp  = 2008-01-15 00:00:00
Not After Timestamp   = 2031-12-31 23:59:59
```

Subject

Name Type = 1 (DN). Rdn attributes=4

```
attributeType = 13 (countryName)           length= 2 data='GB'
attributeType = 9 (organizationName)        length=11 data='Informatica'
attributeType = 10 (organizationalUnitName) length= 9 data='zOS.Admin'
attributeType = 6 (commonName)              length=33
data='STQAtest.mhz890-2.informatica.com'
CN = STQAtest.mhz890-2.informatica.com OU = zOS.Admin O = Informatica C = GB
```

Issuer

Name Type = 1 (DN). Rdn attributes=4

```
attributeType = 13 (countryName)           length= 2 data='GB'
attributeType = 9 (organizationName)        length=11 data='Informatica'
attributeType = 10 (organizationalUnitName) length= 9 data='zOS.Admin'
attributeType = 6 (commonName)              length=32
data='irrcerta.mhz8902.informatica.com'
CN = irrcerta.mhz8902.informatica.com OU = zOS.Admin O = Informatica C = GB
```

Subject Name Attributes

Buffer: length=101

```
data=3063310B300906035504061302474231143012060355040A130B496E666F726D61746963613112301006
0355040B13097A4F532E4164
6D696E312A30280603550403132153545141746573742E6D687A3839302D322E696E666F726D61746963612E6
36F6D
```

```
C = GB
O = Informatica
OU = zOS.Adminca
CN = STQAtest.mhz890-2.informatica.com
```

Issuer Name Attributes

Buffer: length=100

```
data=3062310B300906035504061302474231143012060355040A130B496E666F726D61746963613112301006
0355040B13097A4F532E4164
6D696E312930270603550403132069727263657274612E6D687A383930322E696E666F726D61746963612E636
```

```

F6D
  C = GB
  O = Informatica
  OU = zOS.Adminca
  CN = irrcerta.mhz8902.informatica.com

F51 ValidateCertMode starts
Calling gsk_validate_certificate_mode
Certificate validated OK by gsk_validate_certificate_mode
Issuer_record_id = 1

CA certificates      = 1
Subject certificates = 1
Total certificates   = 2

Terminating
-----

Calling gsk_close_database(). db_handle=1EF7A610. Function address=1F173250
rc=0 X'00000000' from gsk_close_database(). db_handle=0

Run using Job User STQAUPDT <== User does not have authority to TLS-Ring and cannot
create log FILE
-----
-----
PWX-26502 Tracing subtask ended at Wed Jun  6 11:14:18 2018
PWX-00220 DYNALLOCS failed. rc='STQA.V1010SSL.MBAUGRCE.JOB07648.ZO61.N001'. Error
code=X'9700'. Information code=X'0'. File ?.
PWX-33314 TIMEOUTS configuration parameter is deprecated
Processing console program. PWXUGSK CMD=REPORT_CERTIFICATES VERBOSE=Y LOC_TYPE=KEYRING
LOC NAME=ATTLS_keyring
VERBOSE='Y'

Report for command REPORT_CERTIFICATES
=====

Initializing
-----
Calling gsk_get_cms_vector()
Mask = 2047 X'000007FF' (API: LVL1, LVL2, LVL3, LVL4, LVL5, LVL6, LVL7, LVL8, LVL9,
LVL10, LVL11)
Number of functions = 223
gsk_cms_vector. Returned size=892. Compiled size=636

Calling gsk_open_keyring(). Name='ATTLS_keyring'. Function address=1F0ACBF0
rc=x'03353009' (File or keyring not found) from gsk_open_keyring(). Key Ring
Name=ATTLS_keyring
Returning rc=32

```

Ciphers Report

The ciphers report lists the ciphers that are available on the machine where the PowerExchange Listener runs.

To generate a ciphers report, enter the following command:

```
PXUGSK CMD=REPORT_CIPHERS
```

The following output is an example of a ciphers report:

```

PWXUGSK REPORT_CIPHERS
=====
Verbose N
-----
Processing console program. PWXUGSK CMD=REPORT_CIPHERS VERBOSE=N
VERBOSE=''

```

```

Reports for command REPORT CIPHERS
=====
Calling gsk_get_cipher_suites() at address 1EA7C498
gsk_get_cipher_suites() completed OK
Calling gsk_get_all_cipher_suites() at address 1EA51320
gsk_get_all_cipher_suites() completed OK

Building Cipher List Report
-----
Version=4 release=3 security level=1000 (US)
v2_cipher_suites=713642
v3_cipher_suites=050435363738392F303132330A1613100D0915120F0C0306020100
ALL:
v3_cipher_suites=9DA39FA5A13D6A6B686935383936379CA29EA4A03C40673E3F2F323330310A13160D1005
040912150C0F03063B020100

V2    ciphers = 6
V3    ciphers = 48
Total ciphers = 54

V2 Ciphers by Id
=====
Id Name
-- ----
 1 V2 RC4 128-bit with MD5 MAC
 2 V2 RC4 128-bit export with MD5 MAC
 3 V2 RC2 128-bit with MD5 MAC
 4 V2 RC2 128-bit export with MD5 MAC
 6 V2 DES 56-bit with MD5 MAC
 7 V2 Triple DES 168-bit with MD5 MAC

V3 Ciphers by Id
=====
Id Name
-- ----
00 TLS_NULL_WITH_NULL_NULL
01 TLS_NULL_WITH_NULL_MD5
02 TLS_RSA_WITH_NULL_SHA
03 TLS_RSA_WITH_RC4_40_MD5
04 TLS_RSA_WITH_RC4_128_MD5
05 TLS_RSA_WITH_RC4_128_MD5
06 TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5
09 TLS_RSA_WITH_DES_CBC_SHA

V3 Ciphers by Name
=====
Id Name
-- ----
30 TLS_DH_DSS_WITH_AES_128_CBC_SHA
3E TLS_DH_DSS_WITH_AES_128_CBC_SHA256
A4 TLS_DH_DSS_WITH_AES_128_GCM_SHA256
36 TLS_DH_DSS_WITH_AES_256_CBC_SHA
68 TLS_DH_DSS_WITH_AES_256_CBC_SHA256
A5 TLS_DH_DSS_WITH_AES_256_GCM_SHA384
0C TLS_DH_DSS_WITH_DES_CBC_SHA
0D TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA
31 TLS_DH_RSA_WITH_AES_128_CBC_SHA

V3 Ciphers by Key Exchange, Mac
=====
Key Ex  MAC  Encryption  Id Name
-----  ---  -
                                00 TLS_NULL_WITH_NULL_NULL
                                01 TLS_NULL_WITH_NULL_MD5
                                6 V2 DES 56-bit with MD5 MAC
                                3 V2 RC2 128-bit with MD5 MAC
                                4 V2 RC2 128-bit export with MD5 MAC
                                1 V2 RC4 128-bit with MD5 MAC
                                2 V2 RC4 128-bit export with MD5 MAC
                                MD5
                                MD5 DES 56-bit
                                MD5 RC2 128-bit
                                MD5 RC2 128-bit export
                                MD5 RC4 128-bit
                                MD5 RC4 128-bit export

```

	MD5	Triple DES 168-bit	7 V2 Triple DES 168-bit with MD5 MAC
DH_DSS	SHA	AES_128_CBC	30 TLS_DH_DSS_WITH_AES_128_CBC_SHA
DH_DSS	SHA	AES_256_CBC	36 TLS_DH_DSS_WITH_AES_256_CBC_SHA
DH_DSS	SHA	DES_CBC	0C TLS_DH_DSS_WITH_DES_CBC_SHA
DH_DSS	SHA	3DES_EDE_CBC	0D TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA
DH_DSS	SHA256	AES_128_CBC	3E TLS_DH_DSS_WITH_AES_128_CBC_SHA256
DH_DSS	SHA256	AES_128_GCM	A4 TLS_DH_DSS_WITH_AES_128_GCM_SHA256
DH_DSS	SHA256	AES_256_CBC	68 TLS_DH_DSS_WITH_AES_256_CBC_SHA256
DH_DSS	SHA384	AES_256_GCM	A5 TLS_DH_DSS_WITH_AES_256_GCM_SHA384
DH_RSA	SHA	AES_128_CBC	31 TLS_DH_RSA_WITH_AES_128_CBC_SHA
DH_RSA	SHA	AES_256_CBC	37 TLS_DH_RSA_WITH_AES_256_CBC_SHA
DH_RSA	SHA	DES_CBC	0F TLS_DH_RSA_WITH_DES_CBC_SHA
DH_RSA	SHA256	AES_128_CBC	3F TLS_DH_RSA_WITH_AES_128_CBC_SHA256
DH_RSA	SHA256	AES_128_GCM	A0 TLS_DH_RSA_WITH_AES_128_GCM_SHA256
DH_RSA	SHA256	AES_256_CBC	69 TLS_DH_RSA_WITH_AES_256_CBC_SHA256
DH_RSA	SHA384	AES_256_GCM	A1 TLS_DH_RSA_WITH_AES_256_GCM_SHA384
DHE_DSS	SHA	AES_128_CBC	32 TLS_DHE_DSS_WITH_AES_128_CBC_SHA
DHE_DSS	SHA	AES_256_CBC	38 TLS_DHE_DSS_WITH_AES_256_CBC_SHA
DHE_DSS	SHA	DES_CBC	10 TLS_DHE_DSS_WITH_DES_CBC_SHA
DHE_DSS	SHA	DES_CBC	12 TLS_DHE_DSS_WITH_DES_CBC_SHA
DHE_DSS	SHA256	AES_128_CBC	40 TLS_DHE_DSS_WITH_AES_128_CBC_SHA256
DHE_DSS	SHA256	AES_128_GCM	A2 TLS_DHE_DSS_WITH_AES_128_GCM_SHA256
DHE_DSS	SHA256	AES_256_CBC	6A TLS_DHE_DSS_WITH_AES_256_CBC_SHA256
DHE_DSS	SHA256	AES_256_CBC	6B TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
DHE_DSS	SHA384	AES_256_GCM	A3 TLS_DHE_DSS_WITH_AES_256_GCM_SHA384
DHE_RSA	SHA	AES_128_CBC	33 TLS_DHE_RSA_WITH_AES_128_CBC_SHA
DHE_RSA	SHA	AES_256_CBC	39 TLS_DHE_RSA_WITH_AES_256_CBC_SHA
DHE_RSA	SHA	DES_CBC	13 TLS_DHE_RSA_WITH_DES_CBC_SHA
DHE_RSA	SHA	DES_CBC	15 TLS_DHE_RSA_WITH_DES_CBC_SHA
DHE_RSA	SHA	3DES_EDE_CBC	16 TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
DHE_RSA	SHA256	AES_128_CBC	67 TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
DHE_RSA	SHA256	AES_128_GCM	9E TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
DHE_RSA	SHA384	AES_256_GCM	9F TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
RSA	MD5	RC2_CBC_40	06 TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5
RSA	MD5	RC4-128	04 TLS_RSA_WITH_RC4_128_MD5
RSA	MD5	RC4-128	05 TLS_RSA_WITH_RC4_128_MD5
RSA	MD5	RC4-40	03 TLS_RSA_WITH_RC4_40_MD5
RSA	SHA		02 TLS_RSA_WITH_NULL_SHA
RSA	SHA	AES_128_CBC	2F TLS_RSA_WITH_AES_128_CBC_SHA
RSA	SHA	AES_256_CBC	35 TLS_RSA_WITH_AES_256_CBC_SHA
RSA	SHA	DES_CBC	09 TLS_RSA_WITH_DES_CBC_SHA
RSA	SHA	3DES_EDE_CBC	0A TLS_RSA_WITH_3DES_EDE_CBC_SHA
RSA	SHA256		3B TLS_RSA_WITH_NULL_SHA256
RSA	SHA256	AES_128_CBC	3C TLS_RSA_WITH_AES_128_CBC_SHA256
RSA	SHA256	AES_128_GCM	9C TLS_RSA_WITH_AES_128_GCM_SHA256
RSA	SHA256	AES_256_CBC	3D TLS_RSA_WITH_AES_256_CBC_SHA256
RSA	SHA384	AES_256_GCM	9D TLS_RSA_WITH_AES_256_GCM_SHA384

Ciphers by Message Authentication Code, Encryption, Key Exchange

MAC	Encryption	Key Ex	Id Name
---	-----	-----	--
			00 TLS_NULL_WITH_NULL_NULL
MD5			01 TLS_NULL_WITH_NULL_MD5
MD5	DES 56-bit		6 V2 DES 56-bit with MD5 MAC
MD5	RC2 128-bit		3 V2 RC2 128-bit with MD5 MAC
MD5	RC2 128-bit export		4 V2 RC2 128-bit export with MD5 MAC
MD5	RC2_CBC_40	RSA	06 TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5
MD5	RC4_128-bit		1 V2 RC4 128-bit with MD5 MAC
MD5	RC4 128-bit export		2 V2 RC4 128-bit export with MD5 MAC
MD5	RC4-128	RSA	04 TLS_RSA_WITH_RC4_128_MD5
MD5	RC4-128	RSA	05 TLS_RSA_WITH_RC4_128_MD5
MD5	RC4-40	RSA	03 TLS_RSA_WITH_RC4_40_MD5
MD5	Triple DES 168-bit		7 V2 Triple DES 168-bit with MD5 MAC

SHA		RSA	02	TLS_RSA_WITH_NULL_SHA
SHA	AES_128_CBC	DH_DSS	30	TLS_DH_DSS_WITH_AES_128_CBC_SHA
SHA	AES_128_CBC	DH_RSA	31	TLS_DH_RSA_WITH_AES_128_CBC_SHA
SHA	AES_128_CBC	DHE_DSS	32	TLS_DHE_DSS_WITH_AES_128_CBC_SHA
SHA	AES_128_CBC	DHE_RSA	33	TLS_DHE_RSA_WITH_AES_128_CBC_SHA
SHA	AES_128_CBC	RSA	2F	TLS_RSA_WITH_AES_128_CBC_SHA
SHA	AES_256_CBC	DH_DSS	36	TLS_DH_DSS_WITH_AES_256_CBC_SHA
SHA	AES_256_CBC	DH_RSA	37	TLS_DH_RSA_WITH_AES_256_CBC_SHA
SHA	AES_256_CBC	DHE_DSS	38	TLS_DHE_DSS_WITH_AES_256_CBC_SHA
SHA	AES_256_CBC	DHE_RSA	39	TLS_DHE_RSA_WITH_AES_256_CBC_SHA
SHA	AES_256_CBC	RSA	35	TLS_RSA_WITH_AES_256_CBC_SHA
SHA	DES_CBC	DH_DSS	0C	TLS_DH_DSS_WITH_DES_CBC_SHA
SHA	DES_CBC	DH_RSA	0F	TLS_DH_RSA_WITH_DES_CBC_SHA
SHA	DES_CBC	DHE_DSS	10	TLS_DHE_DSS_WITH_DES_CBC_SHA
SHA	DES_CBC	DHE_DSS	12	TLS_DHE_DSS_WITH_DES_CBC_SHA
SHA	DES_CBC	DHE_RSA	13	TLS_DHE_RSA_WITH_DES_CBC_SHA
SHA	DES_CBC	DHE_RSA	15	TLS_DHE_RSA_WITH_DES_CBC_SHA
SHA	DES_CBC	RSA	09	TLS_RSA_WITH_DES_CBC_SHA
SHA	3DES_EDE_CBC	DH_DSS	0D	TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA
SHA	3DES_EDE_CBC	DHE_RSA	16	TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
SHA	3DES_EDE_CBC	RSA	0A	TLS_RSA_WITH_3DES_EDE_CBC_SHA
SHA256		RSA	3B	TLS_RSA_WITH_NULL_SHA256
SHA256	AES_128_CBC	DH_DSS	3E	TLS_DH_DSS_WITH_AES_128_CBC_SHA256
SHA256	AES_128_CBC	DH_RSA	3F	TLS_DH_RSA_WITH_AES_128_CBC_SHA256
SHA256	AES_128_CBC	DHE_DSS	40	TLS_DHE_DSS_WITH_AES_128_CBC_SHA256
SHA256	AES_128_CBC	DHE_RSA	67	TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
SHA256	AES_128_CBC	RSA	3C	TLS_RSA_WITH_AES_128_CBC_SHA256
SHA256	AES_128_GCM	DH_DSS	A4	TLS_DH_DSS_WITH_AES_128_GCM_SHA256
SHA256	AES_128_GCM	DH_RSA	A0	TLS_DH_RSA_WITH_AES_128_GCM_SHA256
SHA256	AES_128_GCM	DHE_DSS	A2	TLS_DHE_DSS_WITH_AES_128_GCM_SHA256
SHA256	AES_128_GCM	DHE_RSA	9E	TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
SHA256	AES_128_GCM	RSA	9C	TLS_RSA_WITH_AES_128_GCM_SHA256
SHA256	AES_256_CBC	DH_DSS	68	TLS_DH_DSS_WITH_AES_256_CBC_SHA256
SHA256	AES_256_CBC	DH_RSA	69	TLS_DH_RSA_WITH_AES_256_CBC_SHA256
SHA256	AES_256_CBC	DHE_DSS	6A	TLS_DHE_DSS_WITH_AES_256_CBC_SHA256
SHA256	AES_256_CBC	DHE_DSS	6B	TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
SHA256	AES_256_CBC	RSA	3D	TLS_RSA_WITH_AES_256_CBC_SHA256
SHA384	AES_256_GCM	DH_DSS	A5	TLS_DH_DSS_WITH_AES_256_GCM_SHA384
SHA384	AES_256_GCM	DH_RSA	A1	TLS_DH_RSA_WITH_AES_256_GCM_SHA384
SHA384	AES_256_GCM	DHE_DSS	A3	TLS_DHE_DSS_WITH_AES_256_GCM_SHA384
SHA384	AES_256_GCM	DHE_RSA	9F	TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
SHA384	AES_256_GCM	RSA	9D	TLS_RSA_WITH_AES_256_GCM_SHA384

V3 Ciphers by Description

=====

Id Description

--

00 No encryption or message authentication and RSA key exchange
01 No encryption with MD5 message authentication and RSA key exchange
02 No encryption with SHA-1 message authentication and RSA key exchange
3B No encryption with SHA-256 message authentication and RSA key exchange
32 128-bit AES encryption with SHA-1 message authentication and ephemeral Diffie-Hellman key exchange signed with a DSA certificate
33 128-bit AES encryption with SHA-1 message authentication and ephemeral Diffie-Hellman key exchange signed with an RSA certificate
30 128-bit AES encryption with SHA-1 message authentication and fixed Diffie-Hellman key exchange signed with a DSA certificate
31 128-bit AES encryption with SHA-1 message authentication and fixed Diffie-Hellman key exchange signed with an RSA certificate

Note: For PowerExchange Listeners on z/OS machines, the report returns all available ciphers, but AT-TLS rules might limit a user to a subset of the available ciphers.

Error Codes Report

The error codes report lists the error codes that appear in the PowerExchanges logs for attempts to establish a secure connection between the PowerExchange Listener and client.

To generate a cipers report, enter the following command:

```
PWXUGSK CMD=REPORT_ERROR_CODES
```

The following output is an example of an error codes report:

```
PWXUGSK REPORT_ERROR_CODES
=====
Verbose N
-----
Processing console program. PWXUGSK CMD=REPORT_ERROR_CODES VERBOSE=N
VERBOSE=''

Reports for command REPORT_ERROR_CODES
=====

Report 1. Definition names and comments
-----
0 OK
1 INVALID_HANDLE
2 API_NOT_AVAILABLE
3 INTERNAL_ERROR
4 INSUFFICIENT_STORAGE
5 INVALID_STATE for the requested operation
6 KEY_LABEL_NOT_FOUND in key database, SAF key ring or PKCS11 token
7 CERTIFICATE_NOT_AVAILABLE. No certificates in database, key ring or PKCS11
8 ERR_CERT_VALIDATION. Validation error in the subject or CA certificate
9 ERR_CRYPTOC. Cryptographic processing error
10 ERR_ASN. ASN processing error
11 ERR_LDAP. LDAP processing error
12 ERR_UNKNOWN_ERROR. An unexpected error has occurred

101 OPEN_CIPHER_ERROR
102 KEYFILE_IO_ERR while reading certificate database
103 KEYFILE_INVALID_FORMAT. Incorrect key database record format
104 KEYFILE_DUPLICATE_KEY_ERR
105 KEYFILE_DUPLICATE_LABEL_ERR
106 BAD_FORMAT_OR_INVALID_PASSWORD. Incorrect key database password
107 KEYFILE_CERTIFICATE_EXPIRED
108 ERR_LOAD_GSKLIB
109 KEYFILE_NO_CA_CERTIFICATES
```

Report 2. Definition names and comments

```
-----
code      hex Gsk Error Description
----      -
1          1 Handle is not valid

3          3 An internal error has occurred
4          4 Insufficient storage is available
5          5 Handle is in the incorrect state
6          6 Key label is not found
7          7 No certificates available
8          8 Certificate validation error
9          9 Cryptographic processing error
10         A ASN processing error
11         B LDAP processing error
12         C An unexpected error has occurred
13         D Size specified for supplied structure is too small
14         E Required gsk_all_cipher_suites structure not supplied

102        66 Error detected while reading certificate database
103        67 Incorrect key database record format
```

106	6A Incorrect key database password
109	6D No certification authority certificates

CHAPTER 25

PWXUMAP - Map List Utility

This chapter includes the following topics:

- [PWXUMAP Utility Overview, 313](#)
- [Supported Operating Systems for the PWXUMAP Utility, 314](#)
- [General Syntax for the PWXUMAP Utility, 314](#)
- [PWXUMAP Commands and Parameters, 314](#)
- [Running the PWXUMAP Utility, 322](#)
- [Examples of PWXUMAP Reports, 322](#)
- [PWXUMAP Utility Usage Notes, 330](#)

PWXUMAP Utility Overview

You can use the PWXUMAP utility to generate reports that list PowerExchange data maps and extraction maps.

For example, you might want to use the utility to list data maps and extraction maps prior to using the DTLURDMO utility to copy maps from one PowerExchange system to another. You can then use the lists to verify that all data maps and extraction maps have been copied to the target system. Also, Informatica Global Customer Support might request that you use the PWXUMAP utility to provide a snapshot of data maps or extraction maps on your system.

The utility can produce the following types of reports:

- List of data maps and extraction maps
- List of source schemas
- List of map lines in comma-delimited format
- Report of CAPXRT or NRDB metadata
- Summary or detailed report of the currently loaded runtime map

Supported Operating Systems for the PWXUMAP Utility

The PWXUMAP utility runs only on Windows computers.

For information about the supported Windows versions, see the *PowerExchange Installation and Upgrade Guide*.

General Syntax for the PWXUMAP Utility

Use the following general syntax to specify control statements for the PWXUMAP utility:

```
PWXUMAP COMMAND=command_name
OUTPUT_FILE=file_name
TYPE={ALL|CRX|DMX}
LOCATION={node_name|local}
[UID=user_name]
[PWD=password|EPWD=encrypted_password]
[AMLIST=access_method_list]
[MAPMASK=map_mask]
[OVERRIDE_CRX_FILE=extraction_maps_file]
[OVERRIDE_DMX_FILE=data_maps_file]
[SCHEMAMASK=schema_mask]
[TABLEMASK=table_mask]
[TIMESTAMPS={Y|N}]
command-specific_parameters
```

PWXUMAP Commands and Parameters

This section describes the commands and available parameters for each command in the PWXUMAP utility syntax.

The PWXUMAP utility has global parameters that apply to all commands, and command-specific parameters that are only available to some of the commands.

PWXUMAP Global Parameters

The PWXUMAP utility supports the following global parameters and options that pertain to all or multiple commands:

CMD=*command_name*

Required. The command that determines the report type to be written to the output file. For more information, see [“Summary of Commands and Parameters” on page 317](#).

OUTPUT_FILE=*file_name*

Required. The name of the output file for the report, such as `my_registrations.txt`. The output file is a standard text file on the Windows system where you run the utility.

TYPE={ALL|CRX|DMX}

Required. The type of map to include in the report. Options are:

- **ALL**. Include both data maps and extraction maps.
- **CRX**. Include only extraction maps.
- **DMX**. Include only data maps.

LOCATION={node_name|local}

The name of the server where the PowerExchange maps file resides. You can specify **local** if the maps file is on the machine where the utility runs, or you can specify the node name of a remote machine where a PowerExchange Listener runs. The node name must be specified in a **NODE** statement in the local **DBMOVER** configuration file. If the location is a remote Listener, and the **SECURITY** statement in the **DBMOVER** configuration file is set to level 1 or 2, a user ID and either a password or an encrypted password are required.

Default is **local**.

UID=user_name

The user name associated with the server specified by the **LOCATION** parameter. The requirement for this parameter depends the value of the **SECURITY** statement in the PowerExchange **DBMOVER** configuration file associated with the remote Listener.

PWD=password

A clear-text password for the user who is specified in the **UID** parameter. If the utility accesses a remote IBM i (i5/OS) or z/OS location, you can enter a valid PowerExchange passphrase instead of a password. Do not specify both the **EPWD** parameter and **PWD** parameter.

EPWD=encrypted_password

An encrypted password for the user who is specified in the **UID** parameter. Do not specify both the **EPWD** parameter and **PWD** parameter. Use **EPWD** if you are not allowed to store passwords in a readable format.

AMLIST=access_method_list

The **AMLIST** parameter filters the results of the **PWXUMAP** utility reports to specific database and file access methods for the **DTLDESCRIBE**, **LISTMAPS**, **LISTSCHEMAS**, and **PRINTMAPLINES** commands.

Use the following parameter values to filter the results for data maps by access method:

Access Method	Value
Adabas	A
DL1	D
ESDS	E
Db2 for z/OS image	G
IDMS	I
KSDS	K
RRDS	N

Access Method	Value
SEQ	S
Tape	T
User	U
Db2 for z/OS unload	W
Datacom	X
Db2 i or Db2 for z/OS database	Z

Use the following parameter values to filter the results for extraction maps by access method:

Access Method	Value
Adabas	A
Db2 i or Db2 for z/OS	B
DL1	D
ESDS	E
MySQL	F
IDMS	I
KSDS	K
Microsoft SQL Server	L
RRDS	N
ODBA	O
Oracle	P
Db2 for Linux, UNIX, and Windows	V
Datacom	X

For example, to filter a report to list IMS data map names, use the following values for the AMLIST parameter:

```
AMLIST=DO
```

To filter a report of data map names to Db2 i and Db2 for z/OS database data maps, use the following values:

```
AMLIST=B
```

There is no default for AMLIST. If you do not include the AMLIST parameter, all access methods are selected for the criteria specified by the rest of the input to the utility.

MAPMASK=map_mask

A mask that matches one or more data-map or extraction-map names. Use the asterisk (*) wildcard to represent all or part of the map name.

OVERRIDE_CRX_FILE=extraction_maps_file

The name of the extraction-maps file that contains the map information, if different than the current extraction-maps file for the PowerExchange Listener specified in the LOCATION parameter.

OVERRIDE_DMX_FILE=data_maps_file

The name of the data-maps file that contains the map information, if different than the current data-maps file for the PowerExchange Listener specified in the LOCATION parameter.

SCHEMAMASK=schema_mask

A pattern that matches one or more schema names. Use the asterisk (*) wildcard to represent all or part of the map name.

TABLEMASK=table_mask

A pattern that matches one or more table names. Use the asterisk (*) wildcard to represent all or part of the map name.

TIMESTAMPS={Y|N}

Whether the report title includes a time stamp.

Default is **Y**.

Summary of Commands and Parameters

This section describes the commands that you can issue to the PWXUMAP utility, a description of the report that each command produces, and the command-specific parameters:

Command	Report Description	Parameters
DTLDESCRIBE	The CAPXRT or NRDB metadata for the selected schemas or maps.	DESCRIBETYPE NULLLITERAL RETLOGINFMSG
LISTMAPS	A list of schema names and associated map names for the schemas and map types selected by the command parameters.	ZOS_LISTMAPS_FILE_INFO ZOS_LISTMAPS_RAW_INFO ZOS_LISTMAPS_TABLE_INFO global parameters, including AMLIST
LISTSCHEMAS	A list of schema names with the number of associated maps for the schemas and map types selected by the command parameters.	ZOS_LISTSCHEMASSTYLE global parameters, except AMLIST
PRINTMAPLINES	The comma-delimited map lines for each map that matches the criteria specified in the command.	global parameters, including AMLIST
PRINTMAPREPORT	The currently loaded runtime map. The report format and contents are determined by the parameters specified with the command.	GET_BULK_MAPNAME_FROM_CCT OVERRIDE_CCT_FILE REPORT_LEVEL global parameters, except AMLIST

DTLDESCRIBE Command

The DTLDESCRIBE command returns CAPXRT or NRDB metadata for the selected schemas or maps. The information can be filtered by access method, schema-name mask, map-name mask, and table-name mask.

Syntax

The command has the following syntax:

```
PWXUMAP COMMAND=DTLDESCRIBE
TYPE={ALL|CRX|DMX}
OUTPUT FILE=file_name
LOCATION={node_name|local}
[UID=user_name]
[PWD=password|EPWD=encrypted_password]
[AMLIST=access_method_list]
[DESCRIBETYPE=metadata_type]
[MAPMASK=map_mask]
[NULLLITERAL=literal]
[OVERRIDE_CRX_FILE=extraction_maps_file]
[OVERRIDE_DMX_FILE=data_maps_file]
[RETLGINFOMSG={Y|N}]
[SCHEMAMASK=schema_mask]
[TABLEMASK=table_mask]
[TIMESTAMPS={Y|N}]
```

Parameters

The following parameters are specific to this command:

DESCRIBETYPE=*metadata_type*

The metadata to include in the report. Only one option can be specified for the DESCRIBETYPE parameter. Options are:

- **COLUMNS.** Include column metadata.
- **CKEYS.** Include foreign-key metadata for IMS sources.
- **DELEMS.** Include Datacom element metadata for Datacom sources.
- **FIELDS.** Include field metadata for relational sources.
- **MAPS.** Include metadata for data maps and extraction maps.
- **PKEYS.** Include primary-key metadata.
- **RECORDS.** Include record metadata for nonrelational sources.
- **SCHEMAS.** Include schema metadata.
- **TABLES.** Include table metadata
- **TPATHS.** Include PATH metadata for IDMS sources.

Default is **SCHEMAS**.

Tip: The MAPS, SCHEMAS, and TABLES options can run unqualified even when the report includes large numbers of data maps or extraction maps. If you specify other DESCRIBETYPE options, use the MAPMASK, SCHEMAMASK, or TABLEMASK parameters to filter the report results and improve the utility performance.

NULLLITERAL=*literal*

A value to use to replace any null columns that are included in the report. Options are:

- **EMPTY.** Set the null literal to an empty string.
- **string.** Enter a string that replaces the null value. The string can be up to 12 characters long.

Default is no replacement value for null columns.

RETLOGINFOMSG={Y|N}

Specifies whether to include messages from the map log files in the report. Options are:

- **Y**. Include logged messages.
- **N**. Do not include logged messages.

Default is **N**.

For descriptions of the global parameters in the syntax, see [“PWXUMAP Global Parameters” on page 314](#).

LISTMAPS Command

The LISTMAPS command returns list of schema names and associated map names for the schemas selected by the command parameters.

Syntax

The LISTMAPS command has the following syntax:

```
PWXUMAP COMMAND=LISTMAPS
TYPE={ALL|CRX|DMX}
OUTPUT_FILE=file_name
LOCATION={node_name|local}
[UID=user_name]
[PWD=password|EPWD=encrypted_password]
[AMLIST=access_method_list]
[MAPMASK=map_mask_pattern]
[OVERRIDE_CRX_FILE=extraction_maps_file]
[OVERRIDE_DMX_FILE=data_maps_file]
[SCHEMAMASK=schema_mask]
[TABLEMASK=table_mask]
[TIMESTAMPS={Y|N}]
[ZOS_LISTMAPS_FILE_INFO={Y|N}]
[ZOS_LISTMAPS_RAW_INFO={Y|N}]
[ZOS_LISTMAPS_TABLE_INFO={Y|N}]
```

Parameters

The following parameters are specific to the command:

ZOS_LISTMAPS_FILE_INFO={Y|N}

Specifies whether the report includes file information from z/OS data maps. Options are:

- **Y**. Include file information from z/OS data maps.
- **N**. Do not include file information from z/OS data maps.

Default is **N**.

ZOS_LISTMAPS_RAW_INFO={Y|N}

Specifies whether to include the raw, comma-delimited file or table information from z/OS data maps in the report. Options are:

- **Y**. Include the raw file or table information from z/OS data maps.
- **N**. Do not include the raw file or table information from z/OS data maps.

Default is **N**.

ZOS_LISTMAPS_TABLE_INFO={Y|N}

Specifies whether to include table information from z/OS data maps in the report. Options are:

- **Y**. Include table information from z/OS data maps.

- **N.** Do not include table information from z/OS data maps.

Default is **N**.

For descriptions of the global parameters in the syntax, see [“PWXUMAP Global Parameters” on page 314](#).

LISTSCHEMAS Command

The LISTSCHEMAS command returns a list of schema names and associated map counts for the schemas that are selected by the command parameters.

Syntax

The command has the following syntax:

```
PWXUMAP COMMAND=LISTSCHEMAS
TYPE={ALL|CRX|DMX}
OUTPUT_FILE=file_name
LOCATION={node_name|local}
[UID=user_name]
[PWD=password|EPWD=encrypted_password]
[MAPMASK=map_mask_pattern]
[OVERRIDE_CRX_FILE=extraction_maps_file]
[OVERRIDE_DMX_FILE=data_maps_file]
[SCHEMAMASK=schema_mask]
[TABLEMASK=table_mask]
[TIMESTAMPS={Y|N}]
[ZOS_LISTSCHEMASSTYLE={SCHEMA|SCHEMAAM|MAPNAME}]
```

Parameters

The following parameters are specific to the command:

ZOS_LISTSCHEMASSTYLE={SCHEMA|SCHEMAAM|MAPNAME}

Limits the number of duplicate rows in the report for schemas and maps that are returned from PowerExchange on z/OS. Options are:

- **SCHEMA.** Include a single row for each schema. Data map counts are not included.
- **SCHEMAAM.** Include a row for each combination of schema and access method. Data map counts are not included.
- **MAPNAME.** Include a row for each map in the schema with data map counts.

Default is **SCHEMA**.

PRINTMAPLINES Command

The comma-delimited format of a map line represents the runtime mapping of nonrelational records to relational tables and columns. The PRINTMAPLINES command returns the comma-delimited map lines for each map selected by the command parameters.

Syntax

The command has the following syntax:

```
PWXUMAP COMMAND=PRINTMAPLINES
TYPE={ALL|CRX|DMX}
OUTPUT_FILE=file_name
LOCATION={node_name|local}
[UID=user_name]
[PWD=password|EPWD=encrypted_password]
[AMLIST=access_method_list]
```



```
[MAPMASK=map_mask_pattern]
[OVERRIDE_CRX_FILE=extraction_maps_file]
[OVERRIDE_DMX_FILE=data_maps_file]
[SCHEMAMASK=schema_mask]
[TABLEMASK=table_mask]
[TIMESTAMPS={Y|N}]
```

Parameters

The PRINTMAPLINES command accepts the global parameters, including AMLIST. For descriptions of the global parameters, see [“PWXUMAP Global Parameters” on page 314](#).

PRINTMAPREPORT Command

The PRINTMAPREPORT command returns information about the current runtime map. The report format and content is determined by the parameters specified with the command.

Syntax

The command has the following syntax:

```
PWXUMAP COMMAND=PRINTMAPREPORT
TYPE={ALL|CRX|DMX}
OUTPUT_FILE=file_name
LOCATION={node_name|local}
[UID=user_name]
[PWD=password|EPWD=encrypted_password]
[GET_BULK_MAPNAME_FROM_CCT={Y|N}]
[MAPMASK=map_mask_pattern]
[OVERRIDE_CCT_FILE=cct_file_name]
[OVERRIDE_CRX_FILE=extraction_maps_file]
[OVERRIDE_DMX_FILE=data_maps_file]
[REPORT_LEVEL={SINGLELINE|SUMMARY|COLUMNS}]
[SCHEMAMASK=schema_mask]
[TABLEMASK=table_mask]
[TIMESTAMPS={Y|N}]
```

Parameters

The following parameters are specific to the command:

GET_BULK_MAPNAME_FROM_CCT={Y|N}

If REPORT_LEVEL=SINGLELINE and TYPE=CRX, this parameter controls whether the registration name for the data map is included in the report. Options are:

- **Y**. Include the registration name.
- **N**. Do not include the registration name.

Default is **N**.

Note: Including the registration names in the report can significantly affect the performance of the utility.

OVERRIDE_CCT_FILE=cct_file_name

The name of the CCT file that contains the capture registration information, if different than the CCT file that is located on the node specified in the LOCATION parameter. For example, if you use the DTLURDMO utility to copy capture registrations to a target, use this parameter to report registration information from the CCT file on the target system.

REPORT_LEVEL={SINGLELINE|SUMMARY|COLUMNS}

Controls the level of detail included in the report for each map. Options are:

- **SINGLELINE.** Reports a single line of information for each map that is included in the report. This information is also included in the SUMMARY and COLUMNS report outputs.
- **SUMMARY.** Reports additional summary information about each map in the report.
- **COLUMNS.** Reports information about each column in the each map selected for the report. A COLUMN report includes the same information as a SUMMARY report and additionally includes detailed information about each column in the selected maps.

Default is **SINGLELINE**.

For descriptions of the global parameters in the syntax, see [“PWXUMAP Global Parameters” on page 314](#).

Running the PWXUMAP Utility

To run the PWXUMAP utility, perform the following steps:

1. On the Windows server where PowerExchange is installed, open a Command Prompt window.
2. Navigate to the directory where the pwxumap executable is located. By default, `pwxumap.exe` is in the root PowerExchange directory.
3. Issue a utility command from the command line. For example:

```
PWXUMAP CMD=LISTMAPS OUTPUT_FILE=C:\\REPORTS\\MYOUTPUT.TXT LOCATION=ZDB1141
UID=MYNAME PWD=my_password TYPE=ALL MAPMASK=*
```

The utility writes the report to the specified output file.

Examples of PWXUMAP Reports

The following examples show the different types of reports that the PWXUMAP utility can produce.

Example 1. DTLDESCRIBE Report

Run the PWXUMAP utility with the DTLDESCRIBE command to report the metadata for the object types and names that match the command options.

The following command produces the example DTLDESCRIBE report:

```
C:\Informatica\PowerExchange\v.r.m PWXUCRGP COMMAND=DTLDESCRIBE OUTPUT_FILE=.\Files\LOCAL
\PWXUMAP_Listmaps.txt TYPE=ALL
LOCATION=SYSB UID=user_name PWD=password
```

Example DTLDESCRIBE report:

```
2019-10-21 16:13:51      PWXUMAP REPORT      VRM '10.4.0 Build DEV_BUILD'      SYSBUSRA1
=====
Command                  : DTLDESCRIBE
Output file               : .\Files\SYSBUSRA1\DtlDescribe.txt
Location                  : SYSBUSRA1
Type                      : ALL
Schema Mask               : *
```

```

Map Mask          : *
Table Mask        : *
Describe Type     : TABLES
Null Literal      : EMPTY

```

```
CAPXRT SQL 'DTLDESCRIBE TABLES,*,,*_*,Y,,,,,Y,'
```

```

Row
Number,qualifier_1,qualifier_2,table_name,type,comments,acc_mth,acc_mths01,acc_mths02,acc
_mths03,acc_mths04,acc_mths05,base_rec
1,ddrvflgsid,,empss01_map1_employee,TABLE,,C,I,Part,"empss01","map1_employee",,EMPLOYEE
2,ddrvflgsid,,idmsqa_stcrss01_student,TABLE,,C,I,Part,"idmsqa","stcrss01_student",,STUD
3,ddrvflgsid,,stunss01_test14_STUDENT_FLAT,TABLE,,C,I,Part,"stunss01","test14_STUDENT_F
LAT",,STUDENT FLAT
4,d2ims9,,dtld0008_COMPLETE_HIERARCHY,TABLE,,C,O,Part,ims9,COMPLETE_HIERARCHY,,NAMEMAST(N
AMESKIL,ADDRESS,PAYROLL)
5,d2ims9,,dtld0009_COMPLETE_HIERARCHY,TABLE,,C,O,Part,ims9,COMPLETE_HIERARCHY,,SKILMAST(S
KILNAME(EXPR,EDUC))
6,d2ims9,,dup testa_ROOT_SEG1,TABLE,,C,O,Part,ims9,ROOT_SEG1,,ROOT(SEG1)
7,d2ims9,,dup testb_ROOT_SEG1,TABLE,,C,O,Part,ims9,ROOT_SEG1,,ROOT(SEG1)
8,d2ims9,,dup testc_ROOT_SEG1,TABLE,,C,O,Part,ims9,ROOT_SEG1,,ROOT(SEG1)
9,d2ims9,,dup test2_ROOT_SEG1,TABLE,,C,O,Part,ims9,dtld002_ROOT_SEG1,,ROOT(SEG1)
10,d2ims9,,dup test5_ROOT_SEG1,TABLE,,C,O,Part,ims9,dtld002_ROOT_SEG1,,ROOT(SEG1)
11,d2ims9,,rvf0001_dtld002_ROOT,TABLE,,C,O,Part,"ims9","dtld002_ROOT",,ROOT
12,d2ims9,,rvf0002_SEG1,TABLE,,C,O,Part,"ims9","SEG1",,SEG1
13,d2ims9,,rvf0005_SEG2,TABLE,,C,O,Part,"ims9","SEG2",,SEG2
14,d2ims9,,rvf0006_dtld002_ROOT_SEG1_SEG2,TABLE,,C,O,Part,ims9,dtld002_ROOT_SEG1_SEG2,,RO
OT(SEG1(SEG2))
15,d2ims9,,rvf0009_dtld003_ROOT,TABLE,,C,O,Part,"ims9","dtld003_ROOT",,ROOT
16,d2ims9,,rvf001_ROOT,TABLE,,C,O,Part,"ims9","dtld002_ROOT",,ROOT
17,d2ims9,,rvf0010_dtld003_SEG1,TABLE,,C,O,Part,"ims9","dtld003_SEG1",,SEG1
18,d2ims9,,rvf0011_dtld003_COMPLETE_HIERARCHY,TABLE,,C,O,Part,ims9,dtld003_COMPLETE_HIERA
RCHY,,ROOT(SEG1(SEG2))
19,d2ims9,,rvf0013_SEG2,TABLE,,C,O,Part,"ims9","SEG2",,SEG2
20,d2ims9,,rvf0015_STUDENT,TABLE,,C,O,Part,"ims9","STUDENT",,STUDENT
21,d2ims9,,rvf0016_CORSECTN,TABLE,,C,O,Part,"ims9","CORSECTN",,CORSECTN
22,d2ims9,,rvf0017_dtld006_COMPLETE_HIERARCHY,TABLE,,C,O,Part,ims9,dtld006_COMPLETE_HIERA
RCHY,,STUDENT(CORSECTN)
23,d2ims9,,rvf0019_NAMEMAST,TABLE,,C,O,Part,"ims9","NAMEMAST",,NAMEMAST
24,d2ims9,,rvf0020_NAMESKIL,TABLE,,C,O,Part,"ims9","NAMESKIL",,NAMESKIL
25,d2ims9,,rvf0021_ADDRESS,TABLE,,C,O,Part,"ims9","ADDRESS",,ADDRESS
26,d2ims9,,rvf0022_PAYROLL,TABLE,,C,O,Part,"ims9","PAYROLL",,PAYROLL
27,d2ims9,,rvf0027_SKILMAST,TABLE,,C,O,Part,"ims9","SKILMAST",,SKILMAST
28,d2ims9,,rvf0028_SKILNAME,TABLE,,C,O,Part,"ims9","SKILNAME",,SKILNAME
29,d2ims9,,rvf0029_EXPR,TABLE,,C,O,Part,"ims9","EXPR",,EXPR
30,d2ims9,,rvf0030_EDUC,TABLE,,C,O,Part,"ims9","EDUC",,EDUC
31,d2ims9,,rvf0037_STUDENT,TABLE,,C,O,Part,"ims9","STUDENT",,STUDENT
32,d2ims9,,rvf0038_CORSECTN,TABLE,,C,O,Part,"ims9","CORSECTN",,CORSECTN
33,d2ims9,,rvf0039_SEG00001,TABLE,,C,O,Part,"ims9","SEG00001",,SEG00001
34,d2ims9,,rvf0040_SEG01002,TABLE,,C,O,Part,"ims9","SEG01002",,SEG01002
35,d2ims9,,rvf0041_SEG01003,TABLE,,C,O,Part,"ims9","SEG01003",,SEG01003
36,d2ims9,,rvf0042_SEG01004,TABLE,,C,O,Part,"ims9","SEG01004",,SEG01004
37,d2ims9,,rvf0043_SEG01005,TABLE,,C,O,Part,"ims9","SEG01005",,SEG01005
38,d2ims9,,rvf0044_SEG00002,TABLE,,C,O,Part,"ims9","SEG00002",,SEG00002
39,d2ims9,,rvf0045_SEG02002,TABLE,,C,O,Part,"ims9","SEG02002",,SEG02002
40,d2ims9,,rvf0046_SEG02003,TABLE,,C,O,Part,"ims9","SEG02003",,SEG02003
41,d2ims9,,rvf0047_SEG00003,TABLE,,C,O,Part,"ims9","SEG00003",,SEG00003
42,d2ims9,,rvf0048_SEG03002,TABLE,,C,O,Part,"ims9","SEG03002",,SEG03002
43,d2ims9,,rvf0049_SEG03003,TABLE,,C,O,Part,"ims9","SEG03003",,SEG03003
44,d2ims9,,rvf0050_SEG00004,TABLE,,C,O,Part,"ims9","SEG00004",,SEG00004
45,d2ims9,,rvf0051_SEG04002,TABLE,,C,O,Part,"ims9","SEG04002",,SEG04002
46,d2ims9,,rvf0052_SEG04003,TABLE,,C,O,Part,"ims9","SEG04003",,SEG04003
47,d2ims9,,rvf0053_SEG00005,TABLE,,C,O,Part,"ims9","SEG00005",,SEG00005
48,d2ims9,,rvf0055_dtld011_STUDENT_TO_SEG01005,TABLE,,C,O,Part,ims9,dtld011_STUDENT_TO_SE
G01005,,STUDENT(SEG00001(SEG01002(SEG01003(SEG01004(SEG01005))))
49,d2ims9,,rvf0061_SEG1,TABLE,,C,O,Part,"ims9","SEG1",,SEG1
50,d2ims9,,rvf0062_ROOT,TABLE,,C,O,Part,"ims9","ROOT",,ROOT
51,d2ims9,,rvf0063_ROOT,TABLE,,C,O,Part,"ims9","ROOT",,ROOT
52,d2ims9,,xxx0001_dtld002_ROOT,TABLE,,C,O,Part,"ims9","dtld002_ROOT",,ROOT
53,d4ada200,,rvf0001_ada200f001_ADA_RECORD,TABLE,,C,A,Part,"ada200","ada200f001_ADA_RECOR

```

```

D",,ADA_RECORD
54,d4ada200,,rvf0002_save_ADA_RECORD,TABLE,,C,A,Part,"ada200","save_ADA_RECORD",,ADA_RECO
RD
55,d4ada8242,,adatemp_ada200f001_ADA_RECORD,TABLE,,C,A,Part,"ada200","ada200f001_ADA_RECO
RD",,ADA_RECORD
56,d4ada8242,,ada8242_map351a_ADA_RECORD,TABLE,,C,A,Part,"ada8242","map351a_ADA_RECORD",,
ADA_RECORD
57,d6cdcfld,,rvf0001_ksds11_t1,TABLE,,C,K,Part,"CDCFIELD","ksds11_t1",,r1
58,d6fldvsam,,cdcfd1_cdcfield1_t1,TABLE,,C,K,Part,"mvs800","cdcfield1_t1",,r1
59,d6fldvsam,,esds13_esds13_t1,TABLE,,C,E,Part,"cdcfield","esds13_t1",,r1
60,d6fldvsam,,ks120001_ksds12_t1,TABLE,,C,K,Part,"cdcfield","ksds12_t1",,r1
61,d6fldvsam,,rvf0001_ksds11_t1,TABLE,,C,K,Part,"CDCFIELD","ksds11_t1",,r1
62,d7muf15q0,,aa_pay_PAYROLL,TABLE,,C,X,Part,"aa","pay_PAYROLL",,PAYROLL
63,d7muf11811,,pwx1877a_map1_PAYROLL,TABLE,,C,X,Part,"dcom_pay","map1_PAYROLL",,PAYROLL
64,d7muf11811,,pwx1877b_map1_PERSONNEL,TABLE,,C,X,Full,"dcom_pmf","map1_PERSONNEL",,PERSO
NNEL
65,d7muf11811,,sfl1997_sfl1997_STUF_SFL,TABLE,,C,X,Full,"dc_sfl","sfl1997_STUF_SFL",,STUF_SF
L
66,d7muf11811,,sfl1997m_sfl1997_STUF997_SFL,TABLE,,C,X,Part,"dcom","sfl1997_STUF997_SFL",,ST
UF997_SFL
67,d7muf11811,,td7crrcm_test11_CRSEM_CRM,TABLE,,C,X,Part,"dc_crsem","test11_CRSEM_CRM",,C
RSEM_CRM
68,d7muf11811,,td7crrcrs_test11_CRSE_CRS,TABLE,,C,X,Part,"dc_crse","test11_CRSE_CRS",,CRSE
_CRS
69,d7muf11811,,td7crsfl_test11_STUF_SFL,TABLE,,C,X,Part,"dc_stuf","test11_STUF_SFL",,STUF
_SFL
70,d7muf11811,,td7crstu_test11_STUW_STU,TABLE,,C,X,Part,"dc_stuw","test11_STUW_STU",,STUW
_STU
71,u4ada8242,,adatemp_ada_record,TABLE,,C,A,Part,"ada200","ada200f001_ADA_RECORD",,ADA_RE
CORD
72,u4ada8242,,ada8242_map351a_ADA_RECORD,TABLE,,C,A,Part,"ada8242","map351a_ADA_RECORD",,
ADA_RECORD

```

```

NRDB SQL 'DTLDESCRIBE TABLES,* ,* ,* ,Y , , , , ,Y , '

```

```

Row
Number,qualifier_1,qualifier_2,table_name,type,comments,acc_mth,acc_mths01,acc_mths02,acc
_mths03,acc_mths04,acc_mths05,base_rec
73,a,notfound,t1,TABLE,,S,c:\temp\notfound.txt,,,,,r1
74,aaada1000,fdt50,YYY_ADA_RECORD,TABLE,,A,1000,50,,,,ADA_RECORD
75,aacfg,map1,t1,TABLE,,S,##DTLCFG,,,,,r1
76,aaksds12,ksds12,t1,TABLE,,K,'USRA1.CDCFIELD.KSDS12',C,,,,r1
77,ada,employee,employee_tab,TABLE,,A,1000,1,,,,employee
78,ada200,ada200f001,ADA_RECORD,TABLE,,A,200,1,,,,ADA_RECORD
79,ada200,save,ADA_RECORD,TABLE,,A,200,1,,,,ADA_RECORD
80,bigims32,map1,BIGROOT,TABLE,,D,BIGIMS32,HDAM,,,,BIGROOT
81,bigims32,map1,BIGCHILD,TABLE,,D,BIGIMS32,HDAM,,,,BIGCHILD
82,bigims32,map1,BIGROOT_TO_BIGCHILD,TABLE,,D,BIGIMS32,HDAM,,,,BIGROOT(BIGCHILD)
83,bigims32,map1tab,bigroot_to_bigchild,TABLE,,D,BIGIMS32,HDAM,,,,BIGROOT(BIGCHILD)
84,cdcfield,esds13,t1,TABLE,,E,'USRA1.CDCFIELD.ESDS13',C,,,,r1
85,cdcfield,ksds11,t1,TABLE,,K,'USRA1.CDCFIELD.KSDS11',C,,,,r1
86,cdcfield,ksds12,t1,TABLE,,K,'USRA1.CDCFIELD.KSDS12',C,,,,r1
87,cleanse,charasc1,t1,TABLE,,S,USRA1.TESTNUM.SRC.CHARASC,,,,,r1
88,copytemp,tomapzos,t1,TABLE,,S,USRA1.copytemp,,,,,r1
89,cr384527,map2,IN_CARR_CLMB_REC,TABLE,,S,/dtlqa2/qadata/idrdas.dat,,,,,IN_CARR_CLMB_REC
90,cr384527,map2,IN_DMERC_CLMB_REC,TABLE,,S,/dtlqa2/qadata/
idrdas.dat,,,,,IN_DMERC_CLMB_REC
91,cr384527,map2,IN_FI_HHA_CLMA_REC,TABLE,,S,/dtlqa2/qadata/
idrdas.dat,,,,,IN_FI_HHA_CLMA_REC
92,cr384527,map2,IN_FI_HOSPC_CLMA_REC,TABLE,,S,/dtlqa2/qadata/
idrdas.dat,,,,,IN_FI_HOSPC_CLMA_REC
93,cr384527,map2,IN_FI_IP_SNF_CLMA_REC,TABLE,,S,/dtlqa2/qadata/
idrdas.dat,,,,,IN_FI_IP_SNF_CLMA_REC
94,cr384527,map2,IN_FI_OP_CMA_REC,TABLE,,S,/dtlqa2/qadata/idrdas.dat,,,,,IN_FI_OP_CMA_REC
95,cr384527,map2,IN_HEADER_REC,TABLE,,S,/dtlqa2/qadata/idrdas.dat,,,,,IN_HEADER_REC
96,cr384527,map2,IN_TRAILER_REC,TABLE,,S,/dtlqa2/qadata/idrdas.dat,,,,,IN_TRAILER_REC
97,datamaps,hex1,t1,TABLE,,S,USRA1.PWXV910.V1.DATAMAPS,,,,,r1
98,dccrse800,test1,CRSE_CRS,TABLE,,X,999,56,2,1,Y,CRS
99,eFld,mv001,t1,TABLE,,S,USRA1.EFields.data(EFLD021d),,,,,r1
100,eFld,m116p,TWrite,TABLE,,S,V:\Work\EFields\datamaps\EFLD120.dat,,,,,r1
101,eFld,m116p,TRead,TABLE,,S,V:\Work\EFields\datamaps\EFLD120.dat,,,,,r1

```

```

102,efld,m116up,TWrite,TABLE,,S,V:\Work\EFields\datamaps\EFLD122.dat,,,,,r1
103,efld,m116up,TRead,TABLE,,S,V:\Work\EFields\datamaps\EFLD122.dat,,,,,r1
104,efld,m116uz,TWrite,TABLE,,S,V:\Work\EFields\datamaps\EFLD126.dat,,,,,r1
105,efld,m116uz,TRead,TABLE,,S,V:\Work\EFields\datamaps\EFLD126.dat,,,,,r1
106,efld,m116vc,TWrite,TABLE,,S,V:\Work\EFields\datamaps\EFLD116.dat,,,,,r1
107,efld,m116vc,TRead,TABLE,,S,V:\Work\EFields\datamaps\EFLD116.dat,,,,,r1
108,efld,m116z,TWrite,TABLE,,S,V:\Work\EFields\datamaps\EFLD124.dat,,,,,r1
109,efld,m116z,TRead,TABLE,,S,V:\Work\EFields\datamaps\EFLD124.dat,,,,,r1
110,efld,m117n16,t1,TABLE,,S,V:\Work\EFields\datamaps\EFLD111d.dat,,,,,r1
111,efld,m117n16u,t1,TABLE,,S,V:\Work\EFields\datamaps\EFLD111d.dat,,,,,r1
112,efld,m117n32,t1,TABLE,,S,V:\Work\EFields\datamaps\EFLD111d.dat,,,,,r1
113,efld,m117n32u,t1,TABLE,,S,V:\Work\EFields\datamaps\EFLD111d.dat,,,,,r1
114,efld,m117n64,t1,TABLE,,S,V:\Work\EFields\datamaps\EFLD111d.dat,,,,,r1
115,efld,m117n64u,t1,TABLE,,S,V:\Work\EFields\datamaps\EFLD111d.dat,,,,,r1
116,efld,m117n8,t1,TABLE,,S,V:\Work\EFields\datamaps\EFLD111d.dat,,,,,r1
117,efld,m117n8u,t1,TABLE,,S,V:\Work\EFields\datamaps\EFLD111d.dat,,,,,r1
118,efld,m118n16,TWrite,TABLE,,S,V:\Work\EFields\datamaps\EFLD136.dat,,,,,r1
119,efld,m118n16,TRead,TABLE,,S,V:\Work\EFields\datamaps\EFLD136.dat,,,,,r1
120,efld,m118n16u,TWrite,TABLE,,S,V:\Work\EFields\datamaps\EFLD138.dat,,,,,r1
121,empss01,map1,DEPARTMENT_HOSPITAL_CLAIM,TABLE,,I,empss01,C,empdemo,,DEPARTMENT(EMPLOY
EE(COVERAGE(HOSPITAL_CLAIM)))
122,empss01,map1,DEPARTMENT_NON_HOSP_CLAIM,TABLE,,I,empss01,C,empdemo,,DEPARTMENT(EMPLOY
EE(COVERAGE(NON_HOSP_CLAIM)))
123,empss01,map1,DEPARTMENT_DENTAL_CLAIM,TABLE,,I,empss01,C,empdemo,,DEPARTMENT(EMPLOYEE
(COVERAGE(DENTAL_CLAIM)))
124,empss01,map1,DEPARTMENT_EMPOSITION,TABLE,,I,empss01,C,empdemo,,DEPARTMENT(EMPLOYEE(E
MPOSITION))
125,empss01,map1,DEPARTMENT_EXPERTISE,TABLE,,I,empss01,C,empdemo,,DEPARTMENT(EMPLOYEE(EX
PERTISE))
126,empss01,map1,DEPARTMENT_STRUCTURE,TABLE,,I,empss01,C,empdemo,,DEPARTMENT(EMPLOYEE(ST
RUCTURE))
127,empss01,map1,DEPARTMENT_STRUCTURE_1,TABLE,,I,empss01,C,empdemo,,DEPARTMENT(EMPLOYEE(
STRUCTURE))
128,empss01,map1,EMPLOYEE_HOSPITAL_CLAIM,TABLE,,I,empss01,C,empdemo,,EMPLOYEE(COVERAGE(H
OSPITAL_CLAIM))
129,empss01,map1,EMPLOYEE_NON_HOSP_CLAIM,TABLE,,I,empss01,C,empdemo,,EMPLOYEE(COVERAGE(N
ON_HOSP_CLAIM))
130,empss01,map1,EMPLOYEE_DENTAL_CLAIM,TABLE,,I,empss01,C,empdemo,,EMPLOYEE(COVERAGE(DEN
TAL_CLAIM))
131,empss01,map1,EMPLOYEE_EMPOSITION,TABLE,,I,empss01,C,empdemo,,EMPLOYEE(EMPOSITION)
132,empss01,map1,EMPLOYEE_EXPERTISE,TABLE,,I,empss01,C,empdemo,,EMPLOYEE(EXPERTISE)
133,empss01,map1,EMPLOYEE_STRUCTURE,TABLE,,I,empss01,C,empdemo,,EMPLOYEE(STRUCTURE)
134,empss01,map1,EMPLOYEE_STRUCTURE_1,TABLE,,I,empss01,C,empdemo,,EMPLOYEE(STRUCTURE)
135,empss01,map1,INSURANCE_PLAN,TABLE,,I,empss01,C,empdemo,,INSURANCE_PLAN
136,empss01,map1,JOB_EMPOSITION,TABLE,,I,empss01,C,empdemo,,JOB(EMPOSITION)
137,empss01,map1,OFFICE_HOSPITAL_CLAIM,TABLE,,I,empss01,C,empdemo,,OFFICE(EMPLOYEE(COVER
AGE(HOSPITAL_CLAIM)))
138,empss01,map1,OFFICE_NON_HOSP_CLAIM,TABLE,,I,empss01,C,empdemo,,OFFICE(EMPLOYEE(COVER
AGE(NON_HOSP_CLAIM)))
139,empss01,map1,OFFICE_DENTAL_CLAIM,TABLE,,I,empss01,C,empdemo,,OFFICE(EMPLOYEE(COVERAG
E(DENTAL_CLAIM)))
140,empss01,map1,employee,TABLE,,I,empss01,C,empdemo,,EMPLOYEE
141,empss01,map1,manages,TABLE,,I,empss01,C,empdemo,,EMPLOYEE(STRUCTURE(EMPLOYEE_1))
142,empss01,map1,dept_manages,TABLE,,I,empss01,C,empdemo,,DEPARTMENT(EMPLOYEE(STRUCTURE(
EMPLOYEE_1(OFFICE))))
143,idmsqa,stcrss01,STUD_CRSE_MEMBERS,TABLE,,I,stcrss01,C,stcrseg,,STUD(CRSE_MEMBERS)
144,idmsqa,stcrss01,CRSE_CRSE_MEMBERS,TABLE,,I,stcrss01,C,stcrseg,,CRSE(CRSE_MEMBERS)
145,idmsqa,stcrss01,student,TABLE,,I,stcrss01,C,stcrseg,,STUD
146,idmsqa,stcrss01,course,TABLE,,I,stcrss01,C,stcrseg,,CRSE
147,idmsqa,stcrss01,stud_crse,TABLE,,I,stcrss01,C,stcrseg,,STUD(CRSE_MEMBERS(CRSE))
148,idmsqa,stcrss01,crse_stud,TABLE,,I,stcrss01,C,stcrseg,,CRSE(CRSE_MEMBERS(STUD))
149,ims9,dtld002,ROOT,TABLE,,O,DTLD002,HDAM,,,ROOT
150,ims9,dtld002,SEG1,TABLE,,O,DTLD002,HDAM,,,SEG1
151,ims9,dtld002,COMPLETE_HIERARCHY,TABLE,,O,DTLD002,HDAM,,,ROOT(SEG1(SEG2))
152,ims9,dtld002,ROOT_TO_SEG1,TABLE,,O,DTLD002,HDAM,,,ROOT(SEG1)
153,ims9,dtld002,SEG2,TABLE,,O,DTLD002,HDAM,,,SEG2
154,ims9,dtld002,ROOT_SEG1_SEG2,TABLE,,O,DTLD002,HDAM,,,ROOT(SEG1(SEG2))
155,ims9,dtld002,ThisIsAVeryLongTableNameWith32Ch,TABLE,,O,DTLD002,HDAM,,,ROOT(SEG1(SEG2
))
156,ims9,dtld002,ROOT_SEG1,TABLE,,O,DTLD002,HDAM,,,ROOT(SEG1)
157,ims9,dtld003,ROOT,TABLE,,O,DTLD003,DEDB,,,ROOT

```

```

158,ims9,dtld003,SEG1,TABLE,,O,DTLD003,DEDB,,,,SEG1
159,ims9,dtld003,COMPLETE_HIERARCHY,TABLE,,O,DTLD003,DEDB,,,,ROOT(SEG1(SEG2))
160,ims9,dtld003,ROOT_TO_SEG1,TABLE,,O,DTLD003,DEDB,,,,ROOT(SEG1)
161,ims9,dtld003,SEG2,TABLE,,O,DTLD003,DEDB,,,,SEG2
162,ims9,dtld003,ROOT_SEG1_SEG2,TABLE,,O,DTLD003,DEDB,,,,ROOT(SEG1(SEG2))
163,ims9,dtld006,STUDENT,TABLE,,O,DTLD006,HIDAM,,,,STUDENT
164,ims9,dtld006,CORSECTN,TABLE,,O,DTLD006,HIDAM,,,,CORSECTN
165,ims9,dtld006,COMPLETE_HIERARCHY,TABLE,,O,DTLD006,HIDAM,,,,STUDENT(CORSECTN)
166,ims9,dtld006,STUDENT_TO_CORSECTN,TABLE,,O,DTLD006,HIDAM,,,,STUDENT(CORSECTN)
167,ims9,dtld008,NAMEMAST,TABLE,,O,DTLD008,HDAM,,,,NAMEMAST
168,ims9,dtld008,NAMESKIL,TABLE,,O,DTLD008,HDAM,,,,NAMESKIL
169,ims9,dtld008,ADDRESS,TABLE,,O,DTLD008,HDAM,,,,ADDRESS
170,ims9,dtld008,PAYROLL,TABLE,,O,DTLD008,HDAM,,,,PAYROLL
171,ims9,dtld008,COMPLETE_HIERARCHY,TABLE,,O,DTLD008,HDAM,,,,NAMEMAST(NAMESKIL,ADDRESS,PA
YROLL)
172,ims9,dtld008,NAMEMAST_TO_NAMESKIL,TABLE,,O,DTLD008,HDAM,,,,NAMEMAST(NAMESKIL)
173,ims9,dtld008,NAMEMAST_TO_ADDRESS,TABLE,,O,DTLD008,HDAM,,,,NAMEMAST(ADDRESS)
174,ims9,dtld008,NAMEMAST_TO_PAYROLL,TABLE,,O,DTLD008,HDAM,,,,NAMEMAST(PAYROLL)
175,ims9,dtld009,SKILMAST,TABLE,,O,DTLD009,HDAM,,,,SKILMAST
176,ims9,dtld009,SKILNAME,TABLE,,O,DTLD009,HDAM,,,,SKILNAME
177,ims9,dtld009,EXPR,TABLE,,O,DTLD009,HDAM,,,,EXPR
178,ims9,dtld009,EDUC,TABLE,,O,DTLD009,HDAM,,,,EDUC
179,ims9,dtld009,COMPLETE_HIERARCHY,TABLE,,O,DTLD009,HDAM,,,,SKILMAST(SKILNAME(EXPR,EDUC)
)
180,ims9,dtld009,SKILMAST_TO_EXPR,TABLE,,O,DTLD009,HDAM,,,,SKILMAST(SKILNAME(EXPR))
181,ims9,dtld009,SKILMAST_TO_EDUC,TABLE,,O,DTLD009,HDAM,,,,SKILMAST(SKILNAME(EDUC))
182,ims9,dtld009,SKILMAST_SKILNAME,TABLE,,O,DTLD009,HDAM,,,,SKILMAST(SKILNAME)
183,ims9,dtld009,SKILMAST_EXPR,TABLE,,O,DTLD009,HDAM,,,,SKILMAST(EXPR)
184,ims9,dtld009,SKILMAST_EDUC,TABLE,,O,DTLD009,HDAM,,,,SKILMAST(EDUC)
185,ims9,dtld011,STUDENT,TABLE,,O,DTLD011,DEDB,,,,STUDENT
186,ims9,dtld011,CORSECTN,TABLE,,O,DTLD011,DEDB,,,,CORSECTN
187,ims9,dtld011,SEG00001,TABLE,,O,DTLD011,DEDB,,,,SEG00001
188,ims9,dtld011,SEG01002,TABLE,,O,DTLD011,DEDB,,,,SEG01002
189,ims9,dtld011,SEG01003,TABLE,,O,DTLD011,DEDB,,,,SEG01003
190,ims9,dtld011,SEG01004,TABLE,,O,DTLD011,DEDB,,,,SEG01004
191,ims9,dtld011,SEG01005,TABLE,,O,DTLD011,DEDB,,,,SEG01005
192,ims9,dtld011,SEG00002,TABLE,,O,DTLD011,DEDB,,,,SEG00002
193,ims9,dtld011,SEG02002,TABLE,,O,DTLD011,DEDB,,,,SEG02002
194,ims9,dtld011,SEG02003,TABLE,,O,DTLD011,DEDB,,,,SEG02003
195,ims9,dtld011,SEG00003,TABLE,,O,DTLD011,DEDB,,,,SEG00003
196,ims9,dtld011,SEG03002,TABLE,,O,DTLD011,DEDB,,,,SEG03002
197,ims9,dtld011,SEG03003,TABLE,,O,DTLD011,DEDB,,,,SEG03003
198,ims9,dtld011,SEG00004,TABLE,,O,DTLD011,DEDB,,,,SEG00004
199,ims9,dtld011,SEG04002,TABLE,,O,DTLD011,DEDB,,,,SEG04002
200,ims9,dtld011,SEG04003,TABLE,,O,DTLD011,DEDB,,,,SEG04003
201,ims9,dtld011,SEG00005,TABLE,,O,DTLD011,DEDB,,,,SEG00005
202,ims9,dtld011,STUDENT_TO_CORSECTN,TABLE,,O,DTLD011,DEDB,,,,STUDENT(CORSECTN)
203,ims9,dtld011,STUDENT_TO_SEG01005,TABLE,,O,DTLD011,DEDB,,,,STUDENT(SEG00001(SEG01002(S
EG01003(SEG01004(SEG01005))))))
204,ims9,dtld011,STUDENT_TO_SEG02003,TABLE,,O,DTLD011,DEDB,,,,STUDENT(SEG00002(SEG02002(S
EG02003)))
205,ims9,dtld011,STUDENT_TO_SEG03003,TABLE,,O,DTLD011,DEDB,,,,STUDENT(SEG00003(SEG03002(S
EG03003)))
206,ims9,dtld011,STUDENT_TO_SEG04003,TABLE,,O,DTLD011,DEDB,,,,STUDENT(SEG00004(SEG04002(S
EG04003)))
207,ims9,dtld011,STUDENT_TO_SEG00005,TABLE,,O,DTLD011,DEDB,,,,STUDENT(SEG00005)

```

Example 2. LISTMAPS report

Run the PWXUMAP utility with the LISTMAPS command to report the schema and map names that match the command options.

The following command produces the example LISTMAPS report:

```

C:\Informatica\PowerExchangev.r.m PWXUCRGP COMMAND=LISTMAPS OUTPUT_FILE=.\Files\LOCAL
\PWXUMAP Listmaps.txt TYPE=ALL
LOCATION=MHZEUSER2 UID=user_name PWD=password

```

Example LISTMAPS report:

```
2019-10-17 11:11:18      PWXUMAP REPORT      VRM '10.4.0'      MHZEUSER2
=====

Command          : LISTMAPS
Output file       : .\Files\LOCAL\PWXUMAP_Listmaps.txt
Type             : ALL
Location         : MHZEUSER2
SchemaMask       : *
MapMask          : *

DMY selection SQL 'DBLIST CRX,*,*, '

Schema           Map
-----
ddrvflgsid      empss01
ddrvflgsid      idmsqa
ddrvflgsid      stunss01

d2ims9          dtld0006
d2ims9          dtld0008
d2ims9          dtld0009
d2ims9          dtld0011
d2ims9          dtld0002
d2ims9          dtld0003

d4ada200        rvf0001
d4ada200        rvf0002

Total CRX Schemas      3
Total CRX Maps          11

DMY selection SQL 'DBLIST DMX,*,*, '

Schema           Map
-----
a                notfound

aaadal000        fdt50

ada200           ada200f001
ada200           save

bigims32         map1

cleanse          charasc1

cpn301           mhzeross

cr384527         map2

datamaps         hex1

efld             mv001

Total DMX Schemas      9
Total DMX Maps          9
```

Example 3. LISTSCHEMAS Report

Run the PWXUMAP utility with the LISTSCHEMAS command to report the schemas that match the command options and the number of maps for each schema.

The following command produces the example LISTSCHEMAS report:

```
C:\Informatica\PowerExchange\v.r.m PWXUCRGP COMMAND=LISTSCHEMAS OUTPUT_FILE=.\Files\LOCAL
\PWXUMAP_ListSchemas.txt TYPE=ALL
LOCATION=MHZEUSER2 UID=user_name PWD=password
SCHEMAMASK=*
```

Example LISTSCHEMAS report:

```
2019-10-17 11:11:18      PWXUMAP REPORT      VRM '10.4.0'      MHZEUSER2
=====
Command      : LISTSCHEMAS
Output file   : .\Files\LOCAL\PWXUMAP_ListSchemas.txt
Type         : ALL
Location      : MHZEUSER2
SchemaMask    : *
MapMask       : *

DMY selection SQL 'DBLIST CRX,*,*, '

Schema      # Maps  DB Type  Instance
-----
ddrvflgsid      3  IDMS      rvflgsid
d2ims9           6  IMS       ims9
d4ada200         2  Adabas   ada200
d4ada8242        2  Adabas   ada8242
d6fldvsam        1  VSAM     fldvsam
d7mufr15q0       1  Datacom  mufr15q0
u4ada8242        2  Adabas   ada8242

Total CRX Schemas      7
Total CRX Maps         17

DMY selection SQL 'DBLIST DMX,*,*, '

Schema      # Maps
-----
a            1
aaada1000    1
ada200       2
bigims32     1
cleanse      1
cpn301       1
cr384527     1
datamaps     1
efld         112
empss01      1
gdgin        1
gdgout       1
general      1
hspv         1
idmsqa       1
imsesds      1
imsksdss     2
imsrrds      2
imsseq       3
imstape      1
ims9         6

Total CRX Schemas      21
Total CRX Maps         142
```

Example 4. PRINTMAPLINES Report

Run the PWXUMAP utility with the PRINTMAPLINES command to report the comma-delimited map lines for the schemas and maps that match the command options.

Important: The comma-delimited map lines in the PRINTMAPLINES report are a proprietary format that can change from release to release. This report is for informational and diagnostic purposes only. To migrate data maps and extraction maps, use the features of the DTLURDMO utility.

The following command produces the example PRINTMAPLINES report:

```
C:\Informatica\PowerExchangev.r.m PWXUCRGP COMMAND=PRINTMAPLINES OUTPUT_FILE=.\Files
\LOCAL\PWXUMAP_Maplines.txt TYPE=CRX
LOCATION=MHZEUSER2 UID=user_name PWD=password
SCHEMAMASK=d4ada200 MAPMASK=rvf0001
```

Example PRINTMAPLINES report:

```
2019-10-17 11:11:18      PWXUMAP REPORT      VRM '10.4.0'      MHZEUSER2
=====
Command                : PRINTMAPLINES
Output file             : .\Files\LOCAL\PWXUMAP_Maplines.txt
Type                   : CRX
Location               : MHZEUSER2
SchemaMask             : d4ada200
MapMask                : rvf0001

DMY selection SQL 'DBLIST CRX,d4ada200,rvf0001,'

CRX map d4ada200.rvf0001
-----
VERSION,10.4.0
FILE,"Data Capture Registration Map",C,rvf0001,,,,,,,,,,,,,D,,19,80,1,1,Y,"*",T,T,,,,,
200,1,,,,,,,,,,,,,Y,A,,,,,,,,,,,,,
T,,ada200f001_ADA_RECORD,
D,ADA_RECORD,N,,,,,
S,"AA",1,1,A,DEU
C,, "DTL__CAPXRESTART1", "ADA_RECORD:DTL__CAPXRESTART1",,,
C,, "DTL__CAPXRESTART2", "ADA_RECORD:DTL__CAPXRESTART2",,,
C,, "DTL__CAPXUOW", "ADA_RECORD:DTL__CAPXUOW",,,
C,, "DTL__CAPXUSER", "ADA_RECORD:DTL__CAPXUSER",,,
C,, "DTL__CAPXTIMESTAMP", "ADA_RECORD:DTL__CAPXTIMESTAMP",,,
C,, "DTL__CAPXACTION", "ADA_RECORD:DTL__CAPXACTION",,,
C,, "PERSONNEL_ID", "ADA_RECORD:PERSONNEL_ID",,,
C,, "FIRST_NAME", "ADA_RECORD:FIRST_NAME",,,
C,, "MIDDLE_I", "ADA_RECORD:MIDDLE_I",,,
C,, "NAME", "ADA_RECORD:NAME",,,
C,, "MAR_STAT", "ADA_RECORD:MAR_STAT",,,
C,, "SEX", "ADA_RECORD:SEX",,,
C,, "BIRTH", "ADA_RECORD:BIRTH",,,
C,, "ADDRESS_LINE_1", "ADA_RECORD:ADDRESS_LINE[1]",,,
C,, "ADDRESS_LINE_2", "ADA_RECORD:ADDRESS_LINE[2]",,,
C,, "ADDRESS_LINE_3", "ADA_RECORD:ADDRESS_LINE[3]",,,
C,, "ADDRESS_LINE_4", "ADA_RECORD:ADDRESS_LINE[4]",,,
C,, "ADDRESS_LINE_5", "ADA_RECORD:ADDRESS_LINE[5]",,,
C,, "CITY", "ADA_RECORD:CITY",,,
C,, "POST_CODE", "ADA_RECORD:POST_CODE",,,
C,, "COUNTRY", "ADA_RECORD:COUNTRY",,,
C,, "AREA_CODE", "ADA_RECORD:AREA_CODE",,,
C,, "PHONE", "ADA_RECORD:PHONE",,,
C,, "DEPT", "ADA_RECORD:DEPT",,,
C,, "JOB_TITLE", "ADA_RECORD:JOB_TITLE",,,
C,, "CURR_CODE_1", "ADA_RECORD:CURR_CODE[1]",,,
C,, "SALARY_1", "ADA_RECORD:SALARY[1]",,,
C,, "BONUS_1_1", "ADA_RECORD:BONUS[1][1]",,,
C,, "BONUS_1_2", "ADA_RECORD:BONUS[1][2]",,,
C,, "BONUS_1_3", "ADA_RECORD:BONUS[1][3]",,,
C,, "BONUS_1_4", "ADA_RECORD:BONUS[1][4]",,,
C,, "BONUS_1_5", "ADA_RECORD:BONUS[1][5]",,,
C,, "CURR_CODE_2", "ADA_RECORD:CURR_CODE[2]",,,
C,, "SALARY_2", "ADA_RECORD:SALARY[2]",,,
C,, "BONUS_2_1", "ADA_RECORD:BONUS[2][1]",,,
C,, "BONUS_2_2", "ADA_RECORD:BONUS[2][2]",,,
C,, "BONUS_2_3", "ADA_RECORD:BONUS[2][3]",,,
C,, "BONUS_2_4", "ADA_RECORD:BONUS[2][4]",,,
C,, "BONUS_2_5", "ADA_RECORD:BONUS[2][5]",,,
C,, "CURR_CODE_3", "ADA_RECORD:CURR_CODE[3]",,,
C,, "SALARY_3", "ADA_RECORD:SALARY[3]",,,
C,, "BONUS_3_1", "ADA_RECORD:BONUS[3][1]",,,
C,, "BONUS_3_2", "ADA_RECORD:BONUS[3][2]",,,
```

```
C,, "BONUS_3_3", "ADA_RECORD:BONUS[3][3]",,,
C,, "BONUS_3_4", "ADA_RECORD:BONUS[3][4]",,,
C,, "BONUS_3_5", "ADA_RECORD:BONUS[3][5]",,,
```

```
Total CRX Schemas      1
Total CRX Maps          1
```

Example 5. PRINTMAPREPORT Report

Run the PWXUMAP utility with the PRINTMAPREPORT command and the REPORT_LEVEL parameter set to SUMMARY to report summary information for the schemas and maps that match the command options.

The following command produces the example PRINTMAPREPORT report:

```
C:\Informatica\PowerExchangev.r.m PWXUCRGP COMMAND=PRINTMAPREPORT OUTPUT_FILE=.\Files
\LOCAL\PWXUMAP_MapReport.txt TYPE=CRX
LOCATION=MHZEUSER2 UID=user_name PWD=password
SCHEMAMASK=d4ada200 MAPMASK=rvf0001 REPORT_LEVEL=SUMMARY
```

Example PRINTMAPREPORT report:

```
2019-10-17 11:11:18      PWXUMAP REPORT      VRM '10.4.0'      MHZEUSER2
=====
Command                  : PRINTMAPREPORT
Output file               : .\Files\LOCAL\PWXUMAP_MapReport.txt
Type                      : CRX
Location                  : MHZEUSER2
SchemaMask                : d4ada200
MapMask                   : rvf0001
Report Level              : SUMMARY

DMY selection SQL 'DBLIST CRX,d4ada200,rvf0001,'

CRX map d4ada200.rvf0001
-----
DB type                   = A (ADABAS)
Bulk Schema               = ada200
Bulk Data Map             = ada200f001
DB id                     = 200
File Number               = 1
Tables                    = 1 (ada200f001_ADA_RECORD)
Records                   = 1 (ADA_RECORD)

Total CRX Schemas        1
Total CRX Maps            1

A Adabas maps             1
C_CAPX maps               1
-----
Total maps                 1
```

PWXUMAP Utility Usage Notes

Review the following considerations when you use the PWXUMAP utility:

- If you run PWXUMAP utility reports against a PowerExchange instance that runs on z/OS, the user ID you specify for the utility must have read access to the schemas and map files on the z/OS system.

- If you run PWXUMAP utility reports against a PowerExchange instance that runs on z/OS, the process takes exclusive locks on DTLCAMAP, DATAMAPS, and CCT files, even when the access to those files is read-only. This might cause access issues if other PowerExchange processes are running concurrently.

CHAPTER 26

PWXUSSL - PowerExchange SSL Reporting Utility

This chapter includes the following topics:

- [PWXUSSL Utility Overview, 332](#)
- [Supported Operating Systems for the PWXUSSL Utility, 333](#)
- [Control Statement Syntax for PWXUSSL Commands, 333](#)
- [PWXUSSL Commands and Parameters, 333](#)
- [Running the PWXUSSL Utility, 337](#)
- [PWXUSSL Utility Reports, 337](#)

PWXUSSL Utility Overview

Use the PWXUSSL utility to generate reports about SSL libraries and certificates on Linux, UNIX, and Windows. You can also determine the validity of certificates that are available to a specified user.

With the PWXUSSL utility, you can complete the following tasks:

- Convert a PKCS12DER format certificate file to the PEM format for use on a Linux, Unix, or Windows machine.
- Verify that a specified PowerExchange user has the authority to view security certificates for the PowerExchange Listener on Linux, Unix, and Windows, that the certificates are current and valid, and that TCP/IP packets can be sent to the PowerExchange Listener.
- Run a certificate report to view information from a certificate chain file. The report can include multiple certificates in a PEM chain file,
- Run a cipher report to view the cipher suites that are available in the OpenSSL cryptographic library. The report includes the hexadecimal codes that you can use to correlate OpenSSL cipher suites to the AT-TLS cipher suites on z/OS.
- Run a return-code or error-code report to view codes from OpenSSL processing and to troubleshoot secure connections.
- Run a configuration report to view the PowerExchange Listener and client SSL configuration and to troubleshoot any issues associated with the PowerExchange DBMOVER configuration files.
- Run a version report to view the version of OpenSSL that was used to build the cryptographic library. On Linux and UNIX, the cryptographic library file is named **libpmpcrypto**. On Windows, the file is named **PMLIBEAY32.DLL**. The report includes the date of the build and compiler settings.

Supported Operating Systems for the PWXUSSL Utility

The PWXUSSL utility runs on computers that have the following operating systems:

- Linux
- UNIX
- Windows

Control Statement Syntax for PWXUSSL Commands

Use the following general syntax to specify control statements for the PWXUSSL utility:

```
PWXUSSL CMD=command_name command-specific parameters
```

PWXUSSL Commands and Parameters

This section describes the commands that you can enter in the CMD statement of the PWXUSSL syntax and the parameters available for each command.

CONVERT_CERT_PKCS12_PEM Command

Use the CONVERT_CERT_PKCS12_PEM command to convert certificates that were created on z/OS in PKCS12DER format to the PEM format that can be used on Linux, Unix, and Windows machines.

If the certificates for the PowerExchange Listener were created on z/OS in the PKCS12DER format, and were transferred by FTP to a Linux, Unix, or Windows system, you can convert the certificate format to the PEM format more commonly used on those systems.

```
pwxussl CMD=CONVERT_CERT_PKCS12_PEM INFILE=pkcs12_file_name [PWD=password]  
[EPWD=encrypted_password]  
OUT_FILE=pem_file_name [OUT_ENCODING={DES3|DES_EDE3_CBC|NONE}] [OUT_PWD=password]  
[OUT_EPWD=encrypted_password]
```

The CONVERT_CERT_PKCS12_PEM command has the following parameters:

INFILE=*pkcs12_file_name*{PWD=*password*|EPWD=*encrypted_password*}

The input file for the conversion. This is a certificate file in the format PKCS12DER. Either a password or an encrypted password is required to open the file.

**OUT_FILE=*pem_file_name* [OUT_ENCODING={DES3|DES_EDE3_CBC|NONE}][OUT_PWD=*password*]
[OUT_EPWD=*encrypted_password*]**

The PEM formatted file that is the output from the conversion. You can optionally specify one of the following encoding formats for the file:

- DES3. Triple DES encryption. This format is the default.

- DES_EDE3_CBC. Triple DES encryption using cypher DES_EDE3_CBC.
- NONE. The output PEM file is not encrypted.

The PEM file also requires a password or encrypted password that can be used to decrypt the file. If you do not specify a password or encrypted password, the utility defaults to the password used for the input file.

PING Command

Use the PING command to verify that a secure connection can be established between the machine from which you issue the command and a PowerExchange Listener on a remote node.

When you issue the PING command, the PWXUSSL utility steps through each phase of a secure connection to verify that the connection will succeed. You can use the information returned from this command to troubleshoot any issues that might occur.

```
pwxussl CMD=PING PING_LOCATION=node_name
[PING_UID=user_name{PING_PWD=password|PING_EPWD=encrypted_password}]
```

The PING command has the following parameters:

PING_LOCATION=*node_name*

The node name of the remote PowerExchange Listener to ping for the secure connection.

PING_UID=*user_name*

A user name that can be used to establish the secure connection. The user must have the authority to view SSL certificates on the remote node specified by PING_LOCATION.

PING_PWD=*password*

The password associated with the user name specified by PING_UID. Specify either a password or an encrypted password to decrypt the certificate files on the remote node.

PING_EPWD=*encrypted_password*

An encrypted password associated with the user name specified by PING_UID. Specify either a password or an encrypted password to decrypt the certificate files on the remote node.

REPORT_CERTIFICATE Command

The REPORT_CERTIFICATE command generates a report that lists information about the security certificate available on a Linux, Unix, and Windows machine where the utility runs.

The command has the following syntax:

```
pwxussl CMD=REPORT_CERTIFICATE INFILE=input_file_name [INFORM={PEM|PKCS12|DER}]
[REPORTFORMAT={OPENSSL|SUMMARY|TEXT|ALL}] [PWD=password] [EPWD=encrypted_password]
```

The REPORT_CERTIFICATE command has the following parameters:

INFILE=*certificate_file_name*

The input file for the certificate report

INFORM={PEM|PKCS12|DER}

The format of the input file. The PWXUSSL utility supports the following certificate file formats:

- PEM.
- PKCS12.
- DER.

If the INFORM parameter is not specified, the utility attempts to access the file using each supported format.

REPORTFORMAT={OPENSSL|SUMMARY|TEXT|ALL}

The output format of the generated report. Specify one of the following options:

- **OPENSSL.**
- **SUMMARY.**
- **TEXT.**
- **ALL.**

Default is **ALL**.

PWD=password

If the input file is encrypted, you must supply a password or encrypted password for the file.

EPWD=encrypted_password

If the input file is encrypted, you must supply a password or encrypted password for the file.

REPORT_CIPHERS Command

The REPORT_CIPHERS command generates a report that lists all of the ciphers available on the machine where the PowerExchange Listener runs.

The command has the following syntax:

```
pwxussl CMD=REPORT_CIPHERS [CIPHER_LIST=list] [CONTEXT_METHOD={TLSV1|TLSV1_1|TLSV1_2|DTLSV1}]
```

The REPORT_CIPHERS command has the following parameters:

CIPHER_LIST=file_name

A file that contains a list of ciphers to search for on the machine where the PowerExchange Listener runs. If no list is specified, the report returns all ciphers available on the machine.

CONTEXT_METHOD={TLSV1|TLSV1_1|TLSV1_2|DTLSV1}

A filter for the report that searches only for those ciphers that support a particular TLS version. You can filter by the following versions:

- **TLSV1.** TLS Version 1.
- **TLSV1_1.** TLS Version 1.1.
- **TLSV1_2.** TLS Version 1.2.
- **DTLSV1.** DTLS Version 1.

If CONTEXT_METHOD is not specified, the report returns ciphers for all security protocols.

REPORT_CODES Command

The REPORT_CODES command generates a report that lists return codes from an attempt to establish a secure connection between a PowerExchange Listener and client.

The command has the following syntax:

```
pwxussl CMD=REPORT_CODES [CODE_TYPE={ALL|CATYPES|VERIFYRC}]
```

The REPORT_CODES command has the following parameter:

CODE_TYPE={ALL|CATYPES|VERIFYRC}

Filters the report results by the type of return code issued during SSL processing. You can filter the report by the following code types:

- **ALL**. All return codes.
- **CATYPES**. Certificate Authority (CA) return codes.
- **VERIFYRC**. Verify peer certificate return codes.

ALL is the default.

REPORT_CONFIG Command

The REPORT_CONFIG command generates a report that describes the security configuration of the machines that participate in a secured connection. You can filter the results to include a specific participant type, such as a PowerExchange client or PowerExchange Listener.

The REPORT_CONFIG command has the following syntax:

```
pwxussl CMD=REPORT_CONFIG [CLIENT_LISTENER_TYPE={ALL|CLIENT|LISTENER}]  
[NAME=node_name]
```

The REPORT_CONFIG command has the following parameters:

CLIENT_LISTENER_TYPE={ALL|CLIENT|LISTENER}

The machine types for which the command returns configuration information. This parameter has the following options:

ALL

The command returns information about both CLIENT and LISTENER mode processing. This is the default.

CLIENT

The command returns information about CLIENT mode processing that involves the NODE statements used when connecting to remote machines.

LISTENER

The command returns information about LISTENER mode processing for those PowerExchange Listeners that accept connections from remote machines.

NAME=node_name

Enter a specific node name or PowerExchange Listener name mask for a remote node. The report returns configuration information for the node you specify, as qualified by the CLIENT_LISTENER_TYPE parameter. If you do not specify the NAME parameter, the report includes all available configuration information for the machine on which you run the utility.

REPORT_ERROR_CODES Command

The REPORT_ERROR_CODES command generates a report that lists error codes returned from SSL processing during an attempt to establish a secure connection between the PowerExchange Listener and client.

The command has the following syntax:

```
pwxussl CMD=REPORT_ERROR_CODES [ERROR_CODE_TYPE={ALL|LIBRARIES|FUNCTIONS|REASONS}]
```

The REPORT_ERROR_CODES command has the following parameters:

ERROR_CODE_TYPE={ALL|LIBRARIES|FUNCTIONS|REASONS}

Filters the report results by the type of return code issued during SSL processing. You can filter the report by the following code types:

- **ALL.** All error codes.
- **LIBRARIES.**
- **FUNCTIONS.**
- **REASONS.**

ALL is the default.

REPORT_VERSION Commands

The **REPORT_VERSION** command generates a report of OpenSSL version and build information.

The command has the following syntax:

```
pwxussl CMD=REPORT_VERSION
```

The **REPORT_CODES** command has no parameters.

Running the PWXUSSL Utility

You can run a PWXUSSL utility command from a command line on a system where PowerExchange is installed.

Navigate to the directory where the `pwxussl` executable is located. By default, this directory is in the PowerExchange installation directory. Then enter `pwxussl` followed by a command and any parameters, as follows:

```
C:\Informatica\PowerExchangev.r.m pwxussl CMD=command_name parameters
```

PWXUSSL Utility Reports

This section describes the reports that you can generate with the PWXUSSL utility.

Certificate Report

The certificate report provides information from a certificate chain file.

To generate a certificate report, enter the following command:

```
C:\Informatica\PowerExchangev.r.m pwxussl CMD=REPORT_CERTIFICATE infile=input_file_name
```

The following output is an example of a certificate report:

```
Processing console program. pwxussl cmd=report_certificate infile=c:\OpenSSL-win32\bin
\PEM\client.pem
REPORT FOR COMMAND REPORT_CERTIFICATE

File contains 1 X509 certificates and 1 subject names.
Certificate 1. Subject Name "/C=AU/ST=Queensland/O=CryptSoft Pty Ltd/CN=Client test cert
(512 bit)"
```

```

Certificate 1. Serial "02". Version "1 (0x0)".
Valid from "2097-06-09 13:57:56" time zone "Z". Valid to "2098-06-09 13:57:56" time zone
"Z".
Certificate has expired
Subject name "/C=AU/ST=Queensland/O=CryptSoft Pty Ltd/CN=Client test cert (512 bit)"
Issuer name "/C=AU/ST=Queensland/O=CryptSoft Pty Ltd/CN=Test CA (1024 bit)"
Signature algorithm "md5WithRSAEncryption"
Public Key algorithm "rsaEncryption". Size 512 bits.

**** START OF RESULT SET FROM API X509_print_ex_fp ****
Certificate:
  Data:
    Version: 1 (0x0)
    Serial Number: 2 (0x2)
    Signature Algorithm: md5WithRSAEncryption
    Issuer: C=AU, ST=Queensland, O=CryptSoft Pty Ltd, CN=Test CA (1024 bit)
    Validity
      Not Before: Jun  9 13:57:56 1997 GMT
      Not After : Jun  9 13:57:56 1998 GMT
    Subject: C=AU, ST=Queensland, O=CryptSoft Pty Ltd, CN=Client test cert (512 bit)
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      RSA Public Key: (512 bit)
        Modulus (512 bit):
          00:bb:6f:e7:94:32:cc:6e:a2:d8:f9:70:67:5a:5a:
          87:bf:be:1a:ff:0b:e6:3e:87:9f:2a:ff:b9:36:44:
          d4:d2:c6:d0:00:43:0d:ec:66:ab:f4:78:29:e7:4b:
          8c:51:08:62:3a:1c:0e:e8:be:21:7b:3a:d8:d3:6d:
          5e:b4:fc:a1:d9
          Exponent: 65537 (0x10001)
      Signature Algorithm: md5WithRSAEncryption
        70:b4:c9:88:ee:81:94:1b:c9:ca:99:fb:50:b2:c0:13:56:21:
        f9:35:14:6d:a0:4c:34:ec:3c:49:a7:f2:df:6a:d1:dd:ae:1a:
        90:07:bd:de:19:d2:f9:58:82:d9:25:79:38:e9:7c:f6:7b:d5:
        8c:49:48:d5:09:26:21:74:ac:6d:7e:55:37:51:1d:80:8e:fd:
        4e:a3:4b:13:35:d7:f3:d3:00:ea:24:d8:ab:2c:db:73:ca:18:
        6c:6a:af:2a:31:3a:cb:cl:7a:c2:3f:7d:55:c4:18:a2:80:54:
        90:49:41:67:67:24:c4:f5:32:b0:85:2e:06:97:06:ed:09:fc:
        52:29
**** END OF RESULT SET FROM API X509_print_ex_fp ****

Private key information MIIBOwIBAAJBALtv55QyzG6i2PlwZlpah7++Gv8L5j6Hnyr/uTZE1NLG0ABDDexm;
q/R4KedLjFEIYjocDui+IXs62NNtXrT8odkCAwEAQAjAbwXq0vJ/+uyEvsNgxLko

```

Ciphers Report

The ciphers report lists the ciphers that are available in the OpenSSL cryptographic library.

To generate a ciphers report, enter the following command:

```
C:\Informatica\PowerExchange\v.r.m pwxussl CMD=REPORT_CIPHERS
```

The following output is an example of a ciphers report:

```
REPORT FOR COMMAND REPORT_CIPHERS
35 available ciphers
```

Ciphers Report for Hex Id, Strength and Version

0 DHE-RSA-AES256-SHA	hex id=39 strength=0081 version=3
1 DHE-DSS-AES256-SHA	hex id=38 strength=0081 version=3
2 AES256-SHA	hex id=35 strength=0081 version=3
3 EDH-RSA-DES-CBC3-SHA	hex id=16 strength=0081 version=3
4 EDH-DSS-DES-CBC3-SHA	hex id=13 strength=0081 version=3
5 DES-CBC3-SHA	hex id=0A strength=0081 version=3
6 DES-CBC3-MD5	hex id=07 strength=0081 version=2
7 DHE-RSA-AES128-SHA	hex id=33 strength=0081 version=3
8 DHE-DSS-AES128-SHA	hex id=32 strength=0081 version=3

9	AES128-SHA	hex	id=2F	strength=0081	version=3
10	IDEA-CBC-SHA	hex	id=07	strength=0041	version=3
11	IDEA-CBC-MD5	hex	id=05	strength=0041	version=2
12	RC2-CBC-MD5	hex	id=03	strength=0041	version=2
13	DHE-DSS-RC4-SHA	hex	id=66	strength=0041	version=3
14	RC4-SHA	hex	id=05	strength=0041	version=3
15	RC4-MD5	hex	id=04	strength=0041	version=3
16	RC4-MD5	hex	id=01	strength=0041	version=2
17	RC4-64-MD5	hex	id=08	strength=0021	version=2
18	EXP1024-DHE-DSS-DES-CBC-SHA	hex	id=63	strength=0012	version=3
19	EXP1024-DES-CBC-SHA	hex	id=62	strength=0012	version=3
20	EXP1024-RC2-CBC-MD5	hex	id=61	strength=0012	version=3
21	EDH-RSA-DES-CBC-SHA	hex	id=15	strength=0021	version=3
22	EDH-DSS-DES-CBC-SHA	hex	id=12	strength=0021	version=3
23	DES-CBC-SHA	hex	id=09	strength=0021	version=3
24	DES-CBC-MD5	hex	id=06	strength=0021	version=2
25	EXP1024-DHE-DSS-RC4-SHA	hex	id=65	strength=0012	version=3
26	EXP1024-RC4-SHA	hex	id=64	strength=0012	version=3
27	EXP1024-RC4-MD5	hex	id=60	strength=0012	version=3
28	EXP-EDH-RSA-DES-CBC-SHA	hex	id=14	strength=000A	version=3
29	EXP-EDH-DSS-DES-CBC-SHA	hex	id=11	strength=000A	version=3
30	EXP-DES-CBC-SHA	hex	id=08	strength=000A	version=3
31	EXP-RC2-CBC-MD5	hex	id=06	strength=000A	version=3
32	EXP-RC2-CBC-MD5	hex	id=04	strength=000A	version=2
33	EXP-RC4-MD5	hex	id=03	strength=000A	version=3
34	EXP-RC4-MD5	hex	id=02	strength=000A	version=2

Ciphers Report for Key Exchange, Encryption, Signature and Message Authentication

0	DHE-RSA-AES256-SHA	Key Ex=DH	Enc=AES (256)	Au=RSA MAC=SHA1	
1	DHE-DSS-AES256-SHA	Key Ex=DH	Enc=AES (256)	Au=DSS MAC=SHA1	
2	AES256-SHA	Key Ex=RSA	Enc=AES (256)	Au=RSA MAC=SHA1	
3	EDH-RSA-DES-CBC3-SHA	Key Ex=DH	Enc=3DES (168)	Au=RSA MAC=SHA1	
4	EDH-DSS-DES-CBC3-SHA	Key Ex=DH	Enc=3DES (168)	Au=DSS MAC=SHA1	
5	DES-CBC3-SHA	Key Ex=RSA	Enc=3DES (168)	Au=RSA MAC=SHA1	
6	DES-CBC3-MD5	Key Ex=RSA	Enc=3DES (168)	Au=RSA MAC=MD5	
7	DHE-RSA-AES128-SHA	Key Ex=DH	Enc=AES (128)	Au=RSA MAC=SHA1	
8	DHE-DSS-AES128-SHA	Key Ex=DH	Enc=AES (128)	Au=DSS MAC=SHA1	
9	AES128-SHA	Key Ex=RSA	Enc=AES (128)	Au=RSA MAC=SHA1	
10	IDEA-CBC-SHA	Key Ex=RSA	Enc=IDEA (128)	Au=RSA MAC=SHA1	
11	IDEA-CBC-MD5	Key Ex=RSA	Enc=IDEA (128)	Au=RSA MAC=MD5	
12	RC2-CBC-MD5	Key Ex=RSA	Enc=RC2 (128)	Au=RSA MAC=MD5	
13	DHE-DSS-RC4-SHA	Key Ex=DH	Enc=RC4 (128)	Au=DSS MAC=SHA1	
14	RC4-SHA	Key Ex=RSA	Enc=RC4 (128)	Au=RSA MAC=SHA1	
15	RC4-MD5	Key Ex=RSA	Enc=RC4 (128)	Au=RSA MAC=MD5	
16	RC4-MD5	Key Ex=RSA	Enc=RC4 (128)	Au=RSA MAC=MD5	
17	RC4-64-MD5	Key Ex=RSA	Enc=RC4 (64)	Au=RSA MAC=MD5	
18	EXP1024-DHE-DSS-DES-CBC-SHA	Key Ex=DH (1024)	Enc=DES (56)	Au=DSS MAC=SHA1	export
19	EXP1024-DES-CBC-SHA	Key Ex=RSA (1024)	Enc=DES (56)	Au=RSA MAC=SHA1	export
20	EXP1024-RC2-CBC-MD5	Key Ex=RSA (1024)	Enc=RC2 (56)	Au=RSA MAC=MD5	export
21	EDH-RSA-DES-CBC-SHA	Key Ex=DH	Enc=DES (56)	Au=RSA MAC=SHA1	
22	EDH-DSS-DES-CBC-SHA	Key Ex=DH	Enc=DES (56)	Au=DSS MAC=SHA1	
23	DES-CBC-SHA	Key Ex=RSA	Enc=DES (56)	Au=RSA MAC=SHA1	
24	DES-CBC-MD5	Key Ex=RSA	Enc=DES (56)	Au=RSA MAC=MD5	
25	EXP1024-DHE-DSS-RC4-SHA	Key Ex=DH (1024)	Enc=RC4 (56)	Au=DSS MAC=SHA1	export
26	EXP1024-RC4-SHA	Key Ex=RSA (1024)	Enc=RC4 (56)	Au=RSA MAC=SHA1	export
27	EXP1024-RC4-MD5	Key Ex=RSA (1024)	Enc=RC4 (56)	Au=RSA MAC=MD5	export
28	EXP-EDH-RSA-DES-CBC-SHA	Key Ex=DH (512)	Enc=DES (40)	Au=RSA MAC=SHA1	export
29	EXP-EDH-DSS-DES-CBC-SHA	Key Ex=DH (512)	Enc=DES (40)	Au=DSS MAC=SHA1	export
30	EXP-DES-CBC-SHA	Key Ex=RSA (512)	Enc=DES (40)	Au=RSA MAC=SHA1	export
31	EXP-RC2-CBC-MD5	Key Ex=RSA (512)	Enc=RC2 (40)	Au=RSA MAC=MD5	export
32	EXP-RC2-CBC-MD5	Key Ex=RSA (512)	Enc=RC2 (40)	Au=RSA MAC=MD5	export
33	EXP-RC4-MD5	Key Ex=RSA (512)	Enc=RC4 (40)	Au=RSA MAC=MD5	export
34	EXP-RC4-MD5	Key Ex=RSA (512)	Enc=RC4 (40)	Au=RSA MAC=MD5	export

Codes Report

The codes report shows certificate authority types and OpenSSL return codes when verification of peer certificates fails. This report shows all available return codes for the code type specified in the command that generated the report.

To generate a codes report, enter the following command:

```
C:\Informatica\PowerExchangev.r.m pwxussl CMD=ERROR_CODES ERROR_CODE_TYPE=ALL
```

The following output is an example of a codes report:

```
Processing console program. pwxussl cmd=report_codes code_type=all
REPORT FOR COMMAND REPORT_CODES
```

```
Report contains the following sections
(1) Certificate Authority Type Codes
(2) Verify Peer Certificate return codes
```

```
Certificate Authority Type Codes from X509_check_ca()
=====
```

```
CA Description
```

```
-- -----
```

```
0  Subject Identification
1  CA1: X509V3 CA
2
3  CA3: Self-signed X509 V1
4  CA4: Key Usage & CertSign bit
5  CA5: Netscape Type extension
```

```
Verify Peer Certificate return codes
=====
```

```
RC Description
```

```
-- -----
```

```
0  X509_V_OK
1  UNSPECIFIED
2  UNABLE_TO_GET_ISSUER_CERT
3  UNABLE_TO_GET_CRL
4  UNABLE_TO_DECRYPT_CERT_SIGNATURE
5  UNABLE_TO_DECRYPT_CRL_SIGNATURE
6  UNABLE_TO_DECODE_ISSUER_PUBLIC_KEY
7  CERT_SIGNATURE_FAILURE
8  CRL_SIGNATURE_FAILURE
9  CERT_NOT_YET_VALID
10 CERT_HAS_EXPIRED
11 CRL_NOT_YET_VALID
12 CRL_HAS_EXPIRED
13 ERROR_IN_CERT_NOT_BEFORE_FIELD
14 ERROR_IN_CERT_NOT_AFTER_FIELD
15 ERROR_IN_CRL_LAST_UPDATE_FIELD
16 ERROR_IN_CRL_NEXT_UPDATE_FIELD
17 OUT_OF_MEM
18 DEPTH_ZERO_SELF_SIGNED_CERT
19 SELF_SIGNED_CERT_IN_CHAIN
20 UNABLE_TO_GET_ISSUER_CERT_LOCALLY
21 UNABLE_TO_VERIFY_LEAF_SIGNATURE
22 CERT_CHAIN_TOO_LONG
23 CERT_REVOKED
24 INVALID_CA
25 PATH_LENGTH_EXCEEDED
26 INVALID_PURPOSE
27 CERT_UNTRUSTED
28 CERT_REJECTED
29 SUBJECT_ISSUER_MISMATCH
30 AKID_SKID_MISMATCH
31 AKID_ISSUER_SERIAL_MISMATCH
32 KEYUSAGE_NO_CERTSIGN
33 UNABLE_TO_GET_CRL_ISSUER
34 UNHANDLED_CRITICAL_EXTENSION
```

```

35 KEYUSAGE_NO_CRL_SIGN
36 UNHANDLED_CRITICAL_CRL_EXTENSION
37 INVALID_NON_CA
38 PROXY_PATH_LENGTH_EXCEEDED
39 KEYUSAGE_NO_DIGITAL_SIGNATURE
40 PROXY_CERTIFICATES_NOT_ALLOWED
41 INVALID_EXTENSION
42 INVALID_POLICY_EXTENSION
43 NO_EXPLICIT_POLICY
44 DIFFERENT_CRL_SCOPE
45 UNSUPPORTED_EXTENSION_FEATURE
46 UNNESTED_RESOURCE
47 PERMITTED_VIOLATION
48 EXCLUDED_VIOLATION
49 SUBTREE_MINMAX
50 APPLICATION_VERIFICATION
51 UNSUPPORTED_CONSTRAINT_TYPE
52 UNSUPPORTED_CONSTRAINT_SYNTAX
53 UNSUPPORTED_NAME_SYNTAX
54 CRL_PATH_VALIDATION_ERROR
56 SUITE_B_INVALID_VERSION
57 SUITE_B_INVALID_ALGORITHM
58 SUITE_B_INVALID_CURVE
59 SUITE_B_INVALID_SIGNATURE_ALGORITHM
60 SUITE_B_LOS_NOT_ALLOWED
61 SUITE_B_CANNOT_SIGN_P_384_WITH_P_256
62 HOSTNAME_MISMATCH
63 EMAIL_MISMATCH
64 IP_ADDRESS_MISMATCH
65 INVALID_CALL
66 STORE_LOOKUP
67 PROXY_SUBJECT_NAME_VIOLATION

```

Configuration Report

The configuration report provides information about the DBMOVER configuration file settings that are in effect when the report is run.

To generate a configuration report, enter the following command:

```
C:\Informatica\PowerExchange\v.r.m pwxussl CMD=REPORT_CONFIG
```

The following output is an example of a configuration report:

```

Processing console program. pwxussl cmd=report_config
REPORT_CONFIG for file name i:\dbmover.cfg
=====

APIs called accepting connections on listener 'PCSSL' on port 16495
-----
1. SSL_CTX_use_certificate_chain_file()
   with 'C:\ssl\pemsExpiring20420426\client.pem'
2. SSL_CTX_set_default_passwd_cb_userdata()
   with '?????'.
3. SSL_CTX_use_PrivateKey_file()
   with 'C:\ssl\pemsExpiring20420426\client.pem'
4. SSL_CTX_load_verify_locations()
   is NOT called.
5. SSL_CTX_set_cipher_list()
   is NOT called.
6. SSL_CTX_set_options(DONT_INSERT_EMPTY_FRAGMENTS).
   is NOT called.
7. SSL_get_verify_result()
   is NOT called.

APIs connecting to node name 'aix16495ssl' on machine 'aix1' port 16495
-----
1. SSL_CTX_use_certificate_chain_file()

```

```

with 'C:\ssl\pemsExpiring20420426\client.pem'
2. SSL_CTX_set_default_passwd_cb_userdata()
with '?????'.
3. SSL_CTX_use_PrivateKey_file()
with 'C:\ssl\pemsExpiring20420426\client.pem'
4. SSL_CTX_load_verify_locations()
is NOT called.
5. SSL_CTX_set_cipher_list()
is NOT called.
6. SSL_CTX_set_options(DONT_INSERT_EMPTY_FRAGMENTS).
is NOT called.
7. SSL_get_verify_result()
is NOT called.

```

APIs connecting to node name 'linux116495' on machine 'linux1' port 16495

```

-----
1. SSL_CTX_use_certificate_chain_file()
with 'C:\ssl\pemsExpiring20420426\client.pem'
2. SSL_CTX_set_default_passwd_cb_userdata()
with '?????'.
3. SSL_CTX_use_PrivateKey_file()
with 'C:\ssl\pemsExpiring20420426\client.pem'
4. SSL_CTX_load_verify_locations()
is NOT called.
5. SSL_CTX_set_cipher_list()
is NOT called.
6. SSL_CTX_set_options(DONT_INSERT_EMPTY_FRAGMENTS).
is NOT called.
7. SSL_get_verify_result()
is NOT called.

```

SSL parameters in file i:\dbmover.cfg'

```

-----
SSL=(PASS=??????,KEY=C:\ssl\pemsExpiring20420426\client.pem,CALIST=C:\ssl
\pemsExpiring20420426\root.pem)
SSL_REQ_CLNT_CERT=N
SSL_REQ_SRVR_CERT=N
SSL_ALLOW_SELFSIGNED=Y
SSL_CONTEXT_METHOD=TLSv1_1

```

Counts and file validations

```

-----
Listeners not using SSL      4
Listeners with SSL=S        1
Listeners total              5

```

```

Nodes not using SSL          21
Nodes with SSL=S             6
Nodes with SSL=Z             2
Nodes total                  29

```

```

SSL KEY file 'C:\ssl\pemsExpiring20420426\client.pem'
SSLUT_GetPEMFileInfo() rc    0 Valid
Ssl Key File Status          0 Can be used in SSL=(KEY=) parameter
Certificates                  2
Private Key types             1

```

Error Codes Report

The error codes report lists error codes returned from SSL processing during an attempt to establish a secure connection between the PowerExchange Listener and client.

To generate an error codes report, enter the following command:

```
C:\Informatica\PowerExchange\v.r.m pwxussl CMD=REPORT_ERROR_CODES CODE_TYPE=ALL
```

The following output is an example of a codes report:

```
Processing console program. pwxussl cmd=report_error_codes code_type=all
```

```
REPORT FOR COMMAND REPORT_ERROR_CODES
```

```
Report contains the following sections
```

- (1) Libraries
- (2) Library Functions
- (3) Library Reasons

```
Libraries
```

```
-----
```

Code	Library name
1 X'01'	unknown library
2 X'02'	system library
3 X'03'	bignum routines

```
Library Functions
```

```
-----
```

```
Functions for 'system library' X'02'
```

1 X'02001000'	fopen()
2 X'02002000'	connect()
3 X'02003000'	getservbyname()
4 X'02004000'	socket()
5 X'02005000'	ioctlsocket()
6 X'02006000'	bind()
7 X'02007000'	listen()
8 X'02008000'	accept()
9 X'02009000'	WSAStartup()
10 X'0200A000'	opendir()
11 X'0200B000'	fread()

```
Functions for 'bignum routines' X'03'
```

100 X'03064000'	BN_BLINDING_convert_ex()
101 X'03065000'	BN_BLINDING_invert_ex()
102 X'03066000'	BN_BLINDING_new()
103 X'03067000'	BN_BLINDING_update()
104 X'03068000'	BN_bn2dec()
105 X'03069000'	BN_bn2hex()
106 X'0306A000'	BN_CTX_new()
107 X'0306B000'	BN_div()
108 X'0306C000'	bn_expand2()
109 X'0306D000'	BN_mod_exp_mont()
110 X'0306E000'	BN_mod_inverse()
111 X'0306F000'	BN_mod_mul_reciprocal()
112 X'03070000'	BN_mpi2bn()
113 X'03071000'	BN_new()
114 X'03072000'	BN_rand()
115 X'03073000'	BN_usub()
116 X'03074000'	BN_CTX_get()
117 X'03075000'	BN_mod_exp_mont_word()
118 X'03076000'	BN_mod_exp2_mont()
119 X'03077000'	BN_mod_lshift_quick()
120 X'03078000'	BN_EXPAND_INTERNAL()
121 X'03079000'	BN_mod_sqrt()
122 X'0307A000'	BN_rand_range()
123 X'0307B000'	BN_exp()
124 X'0307C000'	BN_mod_exp_mont_consttime()
125 X'0307D000'	BN_mod_exp_recp()
126 X'0307E000'	BN_mod_exp_simple()
127 X'0307F000'	BNRAND()
128 X'03080000'	BN_BLINDING_create_param()
129 X'03081000'	BN_CTX_start()
130 X'03082000'	BN_div_recp()
131 X'03083000'	BN_GF2m_mod()
132 X'03084000'	BN_GF2m_mod_exp()
133 X'03085000'	BN_GF2m_mod_mul()
134 X'03086000'	BN_GF2m_mod_solve_quad()

```

135 X'03087000' BN_GF2m_mod_solve_quad_arr()
136 X'03088000' BN_GF2m_mod_sqr()
137 X'03089000' BN_GF2m_mod_sqrt()
138 X'0308A000' BN_div_no_branch()
139 X'0308B000' BN_mod_inverse_no_branch()
145 X'03091000' BN_lshift()
146 X'03092000' BN_rshift()

```

```

Total Libraries = 3
Total Functions = 57

```

Library Reasons -----

Reasons in library 'unknown library' X'01'

```

2 X'01000002' system lib
3 X'01000003' BN lib
4 X'01000004' RSA lib
5 X'01000005' DH lib
6 X'01000006' EVP lib
7 X'01000007' BUF lib
8 X'01000008' OBJ lib
9 X'01000009' PEM lib
10 X'0100000A' DSA lib
11 X'0100000B' X509 lib
13 X'0100000D' ASN1 lib
14 X'0100000E' CONF lib
15 X'0100000F' CRYPTO lib
16 X'01000010' EC lib
20 X'01000014' SSL lib
32 X'01000020' BIO lib
33 X'01000021' PKCS7 lib
34 X'01000022' X509V3 lib
35 X'01000023' PKCS12 lib
36 X'01000024' RAND lib
37 X'01000025' DSO lib
38 X'01000026' ENGINE lib
39 X'01000027' OCSP lib
47 X'0100002F' TS lib
58 X'0100003A' nested asn1 error
59 X'0100003B' bad asn1 object header
60 X'0100003C' bad get asn1 object call
61 X'0100003D' expecting an asn1 sequence
62 X'0100003E' asn1 length mismatch
63 X'0100003F' missing asn1 eos
64 X'01000040' fatal
65 X'01000041' malloc failure
66 X'01000042' called a function you should not call
67 X'01000043' passed a null parameter
68 X'01000044' internal error
69 X'01000045' called a function that was disabled at compile-time

```

Reasons in library 'system library' X'02'

```

1 X'02000001' Operation not permitted
2 X'02000002' No such file or directory
3 X'02000003' No such process
4 X'02000004' Interrupted function call
5 X'02000005' Input/output error
6 X'02000006' No such device or address
7 X'02000007' Arg list too long
8 X'02000008' Exec format error
9 X'02000009' Bad file descriptor
10 X'0200000A' No child processes
11 X'0200000B' Resource temporarily unavailabl
12 X'0200000C' Not enough space
13 X'0200000D' Permission denied
14 X'0200000E' Bad address
15 X'0200000F' Unknown error
16 X'02000010' Resource device
17 X'02000011' File exists

```



```

18 X'02000012' Improper link
19 X'02000013' No such device
20 X'02000014' Not a directory
21 X'02000015' Is a directory
22 X'02000016' Invalid argument
23 X'02000017' Too many open files in system
24 X'02000018' Too many open files
25 X'02000019' Inappropriate I/O control opera
26 X'0200001A' Unknown error
27 X'0200001B' File too large
28 X'0200001C' No space left on device
29 X'0200001D' Invalid seek
30 X'0200001E' Read-only file system
31 X'0200001F' Too many links
32 X'02000020' Broken pipe
33 X'02000021' Domain error
34 X'02000022' Result too large
35 X'02000023' Unknown error
36 X'02000024' Resource deadlock avoided
37 X'02000025' Unknown error
38 X'02000026' Filename too long
39 X'02000027' No locks available
40 X'02000028' Function not implemented
41 X'02000029' Directory not empty
42 X'0200002A' Illegal byte sequence
43 X'0200002B' Unknown error

Reasons in library 'bignum routines' X'03'
Reasons 1 to 69 are the same as for library X'01' 'unknown library'
100 X'03000064' arg2 lt arg3
101 X'03000065' bad reciprocal
102 X'03000066' called with even modulus
103 X'03000067' div by zero
104 X'03000068' encoding error
105 X'03000069' expand on static bignum data
106 X'0300006A' invalid length
107 X'0300006B' not initialized
108 X'0300006C' no inverse
109 X'0300006D' too many temporary variables
110 X'0300006E' input not reduced
111 X'0300006F' not a square
112 X'03000070' p is not prime
113 X'03000071' too many iterations
114 X'03000072' bignum too long
115 X'03000073' invalid range
116 X'03000074' no solution
118 X'03000076' bits too small
119 X'03000077' invalid shift

Total Libraries = 3

Common Reasons = 69
Unique Reasons = 130
Total Reasons = 199

```

Version Report

The version report reports the version of OpenSSL that was used to build the cryptographic library.

To generate a version report, enter the following command:

```
C:\Informatica\PowerExchangev.r.m pwxussl CMD=REPORT_VERSION
```

The following output is an example of a version report:

```

REPORT FOR COMMAND REPORT_VERSION

SSLEAY_VERSION      = OpenSSL 0.9.8a 11 Oct 2005
SSLEAY_BUILT_ON     = built on: Fri Feb 27 23:04:22 2009

```

```
SSLEAY_PLATFORM      = platform: VC-WIN32
SSLEAY_DIR            = OPENSSELDIR: "/usr/local/ssl"

SSLEAY_CFLAGS
compiler: icl  /MD /Ox /O2 /Ob2 /W3 /WX /Gs0 /GF /Gy /nologo -DOPENSSL_SYSNAME_WIN32 -
DWIN32_LEAN_AND_MEAN -DL_ENDIAN -DDSO_WIN32 -D_CRT_SECURE_NO_DEPRECATED -
DOPENSSL_USE_APPLINK -I./Fdout32dll -DOPENSSL_NO_RC5 -DOPENSSL_NO_MDC2 -DOPENSSL_NO_KRB5
```

INDEX

A

A (Add) [77](#)
AMLIST parameter
values [314](#)

B

Batch Registration Utility
DTLUCBRG [112](#)
Before Image column [126](#)

C

Capture Extraction Process Control [135](#)
capture registrations
task flow for suspending and reactivating registrations [264](#)
capture registrations, copying [176](#)
CCATDMP
DTLUCUDB [146](#)
certificate report, PWXUGSK utility [304](#)
certificate report, PWXUSSL utility [337](#), [342](#)
Change Indicator column [126](#)
ciphers report, PWXUGSK utility [307](#)
ciphers report, PWXUSSL utility [338](#)
CMD parameter
PWXUMAP utility [314](#)
codes report, PWXUSSL utility [340](#)
CONDTYPE parameter
DTLUCBRG utility [113](#)
configuration report, PWXUSSL utility [341](#)
control files
createdatamaps utility [25](#)
CONVERT_CERT_PKCS12_PEM command [333](#)
CREATE_ECDSA command
PWXUDMX utility [298](#)
CREATEBICI parameter
DTLUCBRG utility [113](#)
CREATEBICI Parameter [126](#)
createdatamaps utility
command syntax [22](#)
control files [25](#)
examples [57](#)
IMS data maps [54](#)
log file [46](#)
overview [21](#)
REDEFINE statements [49](#)
schema file [29](#)
unavailable data map properties [56](#)
CRGPREFIX parameter
DTLUCBRG utility [113](#)
CRNAME parameter
PWXUCRGP utility [281](#)

D

D (Delete) [76](#)
data maps
creation utility [21](#)
data maps, copying [169](#)
DB2 long names
Restriction with Event Mark Utility [229](#)
DBINFO [147](#)
DBTYPE parameter
DTLUCBRG utility [113](#)
PWXUCRGP utility [281](#)
DECREMENT_FILE_COUNT command
PWXUDMX utility [298](#)
DELETE_ECDSA command
PWXUDMX utility [298](#)
DESCRIBETYPE parameter [318](#)
DFSSTAT
IMS activity report [80](#)
DISPLAY_ECDSA command
PWXUDMX utility [298](#)
DM_COPY statement, DTLURDMO [169](#)
DTLCUIML utility [78](#)
DTLDESCRIBE command [318](#)
DTLIDLC
DTLULCAT parameter file [154](#)
DTLIDLL
DTLULCAT parameter file [154](#)
DTLREXE
DELETE statement [87](#)
PING statement [89](#)
Remote Program Utility [86](#)
SUBMIT statement [90](#)
SYSTEM statement [92](#)
DTLTKNP.TXT [106](#)
DTLUAPPL utility
ADD and MOD statements and parameters [102](#)
connection statement and parameters [101](#)
END APPL statement [105](#)
PRINT APPL example [110](#)
PRINT APPL statement [105](#)
running the utility on i5/OS [105](#)
running the utility on z/OS [106](#)
TKNPARMS member [105](#)
DTLUCBRG
Adabas ADAOPTS parameters [122](#)
DTLUCBRG
CREATEBICI parameter [126](#)
IMS IMSOPTS parameters [122](#)
Multiple sets of parameters [125](#)
MySQL MYSOPTS parameters [122](#)
Oracle ORAOPTS parameters [122](#)
PostgreSQL PGSOPTS parameters [122](#)
Sample Input [125](#), [126](#)
Source Specific Information [125](#)
source-specific parameters [122](#)

- DTLUCBRG (*continued*)
 - SQL Server MSSOPTS parameters [122](#)
 - VSAM VSMOPTS parameters [122](#)
- DTLUCBRG parameters
 - CONDTYPE [113](#)
 - CREATEBICI [113](#)
 - CRGPREFIX [113](#)
 - DBTYPE [113](#)
 - EPWD [113](#)
 - INSTANCE [113](#)
 - LOCATION [113](#)
 - LOCATION_CRG [113](#)
 - LOCATION_DM [113](#)
 - LOCATION_XDM [113](#)
 - OUTPUT [113](#)
 - PWD [113](#)
 - REPLACE [113](#)
 - REPLACEACTIVE [113](#)
 - REUSECRGNAME [113](#)
 - RPTCOLS [113](#)
 - STATUS [113](#)
 - TABLE [113](#)
 - TESTRUN [113](#)
 - UID [113](#)
- DTLUCBRG utility
 - before-image column [113](#)
 - change-indicator column [113](#)
 - example reports [130](#)
- DTLUCDEP [135](#)
- DTLUCSR2
 - Utility scan program for SR2/SR3 records [143](#)
- DTLUCUDB
 - Gathering Diagnostic Information [151](#)
 - Utility [144](#), [154](#)
- DTLULCAT
 - Catalog program [154](#)
- DTLURDMO
 - DM_COPY statement [169](#)
 - global statements [163](#)
 - REG_COPY statement [176](#)
 - XM_COPY statement [185](#)
- DTLUTSK utility
 - commands issued from a LUW command line [214](#)
 - displaying help on Linux, UNIX, or Windows [212](#)
 - FREEALLOC command on z/OS [215](#)
 - LISTALLOC command on z/OS [215](#)
 - LISTLOCATIONS command on z/OS [215](#)
 - LISTTASK command on z/OS [214](#)
 - overview [208](#)
 - security on z/OS [217](#)
 - STOPTASK command on z/OS [214](#)
 - syntax and parameters on i5/OS [208](#)
 - syntax and parameters on Linux, UNIX, and Windows [210](#)
- DUMP_ECSA command
 - PWXUDMX utility [298](#)
- DUMPDIAG [147](#)

E

- E (ET/BT Record Extraction) [77](#)
- ECSA memory
 - PWXUDMX utility [296](#)
- EDMXLUTL
 - DB2 Long name restrictions [229](#)
- Encrypted password
 - DTLUCBRG utility [113](#)

- EPWD parameter
 - DTLUCBRG utility [113](#)
 - PWXUCRGP utility [281](#)
 - PWXUMAP utility [314](#)
- error codes report, PWXUGSK utility [311](#)
- Event Mark Utility
 - DB2 long names restriction [229](#)
- extraction maps, copying [176](#), [185](#)

F

- FILE_TYPE
 - dtlulcat parameter [154](#)

G

- GET_BULK_MAPNAME_FROM_CCT parameter [321](#)
- global statements, DTLURDMO [163](#)

H

- HELP
 - DTLUCUDB [148](#)

I

- I (Insert) [76](#)
- i5/OS
 - Running DTLUCBRG [127](#)
- IDMS_VERSION
 - dtlulcat parameter [154](#)
- INCREMENT_FILE_COUNT command
 - PWXUDMX utility [299](#)
- INSTANCE parameter
 - DTLUCBRG utility [113](#)
 - PWXUCRGP utility [281](#)
- INSTANCE_IDENTIFIER
 - DTLUCUDB parameter [154](#)

L

- L (Reset Latest Sequence Number) [76](#)
- Linux and UNIX [106](#)
- Linux, UNIX, and Windows
 - Running DTLUCBRG [128](#)
- LISTMAPS command [319](#)
- LISTSCHEMAS command [320](#)
- Local Mode
 - Adding log restrictions [158](#)
- LOCATION parameter
 - DTLUCBRG utility [113](#)
 - PWXUCRGP utility [281](#)
 - PWXUMAP utility [314](#)
- LOCATION_CRG parameter
 - DTLUCBRG utility [113](#)
- LOCATION_DM parameter
 - DTLUCBRG utility [113](#)
- LOCATION_XDM parameter
 - DTLUCBRG utility [113](#)
- Log Catalog
 - Adding Logs in Order [158](#)
- log records
 - user-defined [80](#)

LOGPRT
DTLUCUDB [148](#)

M

MAPMASK parameter
PWXUMAP [314](#)
MEDIA_CONTENT
dtlulcat parameter [154](#)
MEDIA_TYPE
dtlulcat parameter [154](#)
MySQL DDL catalog tables
PWXCATMY utility [237](#)

N

NULLLITERAL parameter [318](#)

O

Operational procedures
Adding logs to the catalog [158](#)
OS/390
Running DTLUCBRG [128](#)
OUTPUT parameter
DTLUCBRG utility [113](#)
OUTPUT_FILE parameter
PWXUCRGP utility [281](#)
PWXUMAP utility [314](#)
OVERRIDE_CCT_FILE parameter
PWXUCRGP utility [281](#)
PWXUMAP utility [314](#)
OVERRIDE_CRX_FILE parameter
PWXUMAP utility [314](#)
OVERRIDE_DMX_FILE parameter
PWXUMAP utility [314](#)
overview [99](#)

P

P (Populate PCAT Control File) [76](#)
PING command for PWXUSL [301](#), [334](#)
PowerExchange Logger for Linux, UNIX, and Windows
PWXUCCLPRT utility [241](#)
PWXUCDCT utility [247](#)
PowerExchange utilities
overview [16](#)
PRINTMAPLINES command [320](#)
PRINTMAPREPORT command [321](#)
PWD parameter
DTLUCBRG utility [113](#)
PWXUCRGP utility [281](#)
PWXUMAP utility [314](#)
PWXCATMY utility
operating systems [238](#)
operation types [238](#)
overview [237](#)
PWXUCCLPRT
CSV output format [244](#)
dump output format [245](#)
examples [244](#)
parameters [242](#)
tokens output [246](#)

PWXUCCLPRT utility
command syntax [242](#)
INPUT statement [242](#)
operating systems [241](#)
OUTPUT statement [243](#)
parameter file example [244](#)
parameters [242](#), [243](#)
PWXUCCLPRT utility
overview [241](#)
PWXUCDCT utility
backing up the CDCT file [255](#)
deleting expired CDCT records [258](#)
deleting orphan CDCT records [257](#)
operating systems [248](#)
overview [247](#)
parameters on commands [253](#)
printing the CDCT file contents [259](#)
re-creating the CDCT file from log files [256](#)
restoring the CDCT file [256](#)
running [254](#)
summary of commands [249](#)
syntax for commands [248](#)
usage notes [255](#)
PWXUCREG utility
examples of utility jobs [276](#)
general syntax for commands [265](#)
global SET_CONTROL_VALUE parameters [271](#)
overview [262](#)
registration-specific command parameters [274](#)
running a utility job [276](#)
summary of commands [266](#)
supported operating system [264](#)
task flow for suspending and reactivating registrations [264](#)
usage considerations [263](#)
PWXUCRGP parameters
CRNAME [281](#)
DBTYPE [281](#)
EPWD [281](#)
INSTANCE [281](#)
LOCATION [281](#)
OUTPUT_FILE [281](#)
OVERRIDE_CCT_FILE [281](#)
PWD [281](#)
REPORT_SEQUENCE [281](#)
TRACING [281](#)
UID [281](#)
PWXUCRGP utility
operating systems [280](#)
overview [280](#)
running [284](#)
syntax for commands [281](#)
PWXUDMX job
to run the PWXUDMX utility [297](#)
PWXUDMX utility
commands [297](#)
CREATE_ECSCA command [298](#)
DECREMENT_FILE_COUNT command [298](#)
DELETE_ECSCA command [298](#)
DISPLAY_ECSCA command [298](#)
DUMP_ECSCA command [298](#)
INCREMENT_FILE_COUNT command [299](#)
operating systems [296](#)
overview [296](#)
running on z/OS [297](#)
PWXUGSK utility
certificate report [304](#)
ciphers report [307](#)
commands [301](#)

PWXUGSK utility (*continued*)

- error codes report [311](#)
- operating systems [300](#)
- overview [300](#)
- parameters [301](#)
- PING command [301](#)
- REPORT_CERTIFICATES command [301](#)
- REPORT_CIPHERS command [302](#)
- REPORT_ERROR_CODES command [302](#)
- syntax for commands [301](#)

PWXUMAP parameters

- CMD [314](#)
- EPWD [314](#)
- GET_BULK_MAPNAME_FROM_CCT [314](#)
- LOCATION [314](#)
- MAPMASK [314](#)
- OUTPUT_FILE [314](#)
- OVERRIDE_CCT_FILE [314](#)
- OVERRIDE_CRX_FILE [314](#)
- OVERRIDE_DMX_FILE [314](#)
- PWD [314](#)
- REPORT_LEVEL [314](#)
- SCHEMAMASK [314](#)
- TIMESTAMPS [314](#)
- TYPE [314](#)
- UID [314](#)

PWXUMAP utility

- DESCRIBETYPE parameter [318](#)
- OVERRIDE_CCT_FILE parameter [321](#)
- RETLOGINFOMSG parameter [318](#)
- ZOS_LISTMAPS_TABLE_INFO [319](#)
- AMLIST parameter [314](#)
- commands [314](#), [317](#)
- DTLDESCRIBE command [318](#)
- GET_BULK_MAPNAME_FROM_CCT parameter [321](#)
- LISTMAPS command [319](#)
- LISTSCHEMAS command [320](#)
- NULLLITERAL parameter [318](#)
- operating systems [314](#)
- parameters [314](#)
- PRINTMAPLINES command [320](#)
- PRINTMAPREPORT command [321](#)
- REPORT_LEVEL parameter [321](#)
- running [322](#)
- ZOS_LISTMAPS_FILE_INFO parameter [319](#)
- ZOS_LISTMAPS_RAW_INFO parameter [319](#)
- ZOS_LISTSCHEMASSTYLE parameter [320](#)

PWXUSSL utility

- certificate report [337](#)
- ciphers report [338](#)
- commands [333](#)
- configuration report [341](#)
- CONVERT_CERT_PKCS12_PEM command [333](#)
- error codes report [342](#)
- operating systems [333](#)
- overview [332](#)
- parameters [333](#)
- PING command [334](#)
- REPORT_CERTIFICATE command [334](#)
- REPORT_CIPHERS command [335](#)
- REPORT_CODES command [335](#)
- REPORT_CONFIG command [336](#)
- REPORT_ERROR_CODES command [336](#)
- REPORT_VERSION command [337](#)
- running [337](#)
- syntax for commands [333](#)
- version report [345](#)

R

- R (Report on PCAT Control File) [76](#)
- REDEFINE statements
 - createdatmaps utility [49](#)
- REG_COPY statement, DTLURDMO [176](#)
- registrations, copying [176](#)
- Remote Execution Utility, DTLREXE
 - DELETE statement [87](#)
 - PING statement [89](#)
 - SUBMIT statement [90](#)
 - SYSTEM statement [92](#)
- REPLACE parameter
 - DTLUCBRG utility [113](#)
- REPLACEACTIVE parameter
 - DTLUCBRG utility [113](#)
- Replacing active registrations
 - DTLUCBRG utility [113](#)
- Replacing registrations
 - DTLUCBRG utility [113](#)
- REPORT_CERTIFICATE command [334](#)
- REPORT_CERTIFICATES command [301](#)
- REPORT_CIPHERS command [302](#), [335](#)
- REPORT_CODES command [335](#)
- REPORT_CONFIG command [336](#)
- REPORT_ERROR_CODES command [302](#), [336](#)
- REPORT_LEVEL parameter
 - PWXUMAP [314](#)
- REPORT_SEQUENCE parameter
 - PWXUCRGP utility [281](#)
- REPORT_VERSION command [337](#)
- RETLOGINFOMSG parameter [318](#)
- REUSECRGNAME parameter
 - DTLUCBRG utility [113](#)
- RPTCOLS parameter
 - DTLUCBRG utility [113](#)
- Running DTLUCBRG
 - Windows and OS/390 [82](#), [127](#), [233](#)

S

- S (Submit ADASEL) [77](#)
- Sample Input
 - DTLUCBRG [125](#), [126](#)
- schema file
 - createdatmaps utility [29](#)
- SCHEMAMASK parameter
 - PWXUMAP [314](#)
- SETDEF
 - DTLUCUDB [149](#)
- SNAPSHOT
 - DTLUCUDB [150](#)
- SR2OUT
 - DTLUCSR2 DD Card [143](#)
- SR2TOTAL
 - DTLUCSR2 DD Card [143](#)
- STATUS parameter
 - DTLUCBRG utility [113](#)

T

- T (Submit ET Record Extraction) [77](#)
- TABLE parameter
 - DTLUCBRG utility [113](#)
- TESTRUN parameter
 - DTLUCBRG utility [113](#)

TIMESTAMPS parameter
 PWXUMAP [314](#)
TRACING parameter
 PWXUCRGP [281](#)
TYPE parameter
 PWXUMAP utility [314](#)

U

UID parameter
 DTLUCBRG utility [113](#)
 PWXUCRGP utility [281](#)
 PWXUMAP utility [314](#)
User-defined log records [80](#)
utilities
 DTLCUIML [78](#)

V

V (Rebuild the PCAT Control File) [76](#)
version report, PWXUSSL utility [345](#)

X

XM_COPY statement, DTLURDMO [185](#)

Z

ZOS_LISTMAPS_FILE_INFO parameter [319](#)
ZOS_LISTMAPS_RAW_INFO parameter [319](#)
ZOS_LISTMAPS_TABLE_INFO [319](#)
ZOS_LISTSCHEMASSTYLE parameter [320](#)